Network Working Group
<draft-bonachea-sftp-00.txt>
Updates: RFC 2228
Internet-Draft
Expires in January, 2000

D. Bonachea
S. McPeak

June, 1999

Protocol Negotiation Extensions to Secure FTP

Status of this Memo

Abstract

This document presents refinements to RFC 2228, "FTP Security Extensions" (October 1997) [1]. It lends support for more efficient and secure protocol negotiation in the presence of multiple protocol possibilities, by adding one new optional command and several new reply codes.

The following new optional command is introduced in this specification:

DIGT (Protocol Negotiation Digest)

This document also seeks to solidify several issues that were left underspecified in RFC 2228. Specifically, it adds additional reply codes to cover cases not mentioned in the original specification and it defines a legal naming convention for security mechanisms.

1. Introduction

Recently, the authors implemented the RFC 2228 protocol, FTP Security Extensions [1]. This protocol extends the basic File Transfer Protocol (FTP) specified in RFC 959 [2], adding security services including authentication and privacy. We assume the reader is familiar with RFC 2228.

During the implementation of RFC 2228 we found it necessary and useful to make several changes to the protocol. This memo details those changes.

Section 2 describes changes which extend RFC 2228, to improve security and efficiency. Section 3 closes specification holes in RFC 2228 itself. Section 4 lifts a RFC 959 restriction, thereby improving total system security.

2. Protocol Negotiation Improvements

DIGT (Protocol Negotiation Digest) command

This new command provides a way for a client to verify that an attacker didn't interfere with the protocol negotiation sequence. This is important because a client that supports multiple security mechanisms might be "tricked" into using its weakest security mechanism by an attacker who interferes with protocol negotiation (i.e. by modifying the appropriate mechanism acceptance replies to look like refusals). The client may issue this optional command at any time after authentication is complete, and the server will reply with a secure hash digest computed over the entire protocol negotiation exchange. The client can then compare this digest to the locally computed digest to verify that both server and client experienced the same protocol negotiation sequence (i.e. there was no outside interference). The hash function is Sha1, and is computed over the entire negotiation sequence from the first character in the first client command (usually an AUTH), to the last character in the server reply which indicates the security data exchange is complete (the LF character on a 234 or 235 reply). The form of the server reply to DIGT will be:

```
211 DIGT=base64data
   ; base64data is a placeholder for the actual digest block,
   ; which must be encoded using the base64 format, as
   ; described in RFC 2228 (see [1] for details).
```

The server may also send a 502 response if the DIGT command is not implemented.

Note the efficacy of this security measure depends on the client asking for the digest faster than an attacker can compromise the current (weaker) security mechanism and substitute a modified digest.

Efficiency improvements to protocol negotiation

RFC 2228 allows for the possibility of clients and servers that implement a large number of security mechanisms, yet it provides no efficient means for the client to discover and request a supported mechanism. Under the RFC 2228 specification, the client repeatedly issues AUTH commands for each security mechanism it supports (in order of preference) until finding one the server will accept. For clients that implement a large number of mechanisms, this could potentially lead to an unreasonably large number of network round trips in order to complete the protocol negotiation sequence. In order to alleviate this problem, this specification adds two new optional server reply codes that allow the server to reveal a list of acceptable security mechanisms in a single reply. This enables the client to perform the search locally and select an acceptable mechanism in a single network roundtrip.

The new reply codes to the AUTH command are:

504 Security mechanism unrecognized. [LIST=mechlist]
504 Security mechanism unrecognized. [ILST=mechlist]

Here, mechlist is a space-delimited list of security mechanisms that the server understands and might accept from this client. The keyword LIST indicates that mechlist is a complete list of acceptable security mechanisms, and any mechanisms not listed will be refused. The keyword ILST indicates that mechlist is an incomplete list of acceptable security mechanisms, and that other acceptable unlisted security mechanisms might exist. The square brackets are not to be included in the actual reply, but indicate that the mechanism list is optional.

3. Amendments to RFC 2228

New server replies

This section presents several new server reply codes which were missing in the original specification.

```
234 Security data exchange complete. [ADAT=base64data]
 ;  This reply to AUTH indicates the security data exchange
 ;  completed successfully.  The square brackets are not to be
 ;  included in the reply, but indicate that security data in
 ;  the reply is optional.
```

This reply code existed in RFC 2228, but did not allow for the possibility of the server sending security data (for protocols where the only data exchange necessary is for the server to send a block to the client).

```
503 Reauthentication is prohibited.
 ;  This reply code to an AUTH indicates that a successful
 ;  security data exchange has completed, and this server is
 ;  unwilling to begin a new authentication session (the
 ;  client must "hang up" and reconnect)
```

RFC 2228 specifies that a server may prohibit the client from reissuing the AUTH command in order to establish new authentication, but provides no reply code to communicate this.

PBSZ Negotiation

During PBSZ (Protocol Buffer Size) negotiation, the client proposes a buffer size, and the server may accept this size, reply with a smaller size, or give an error. The size agreed upon is subsequently used during data transfers as a maximum bound on the size of each encrypted block. However, some protocol mechanisms have a minimum encrypted block size, and RFC 2228 doesn't provide a way for the server to indicate that the proposed block size is too small. To remedy this, this specification adds a new reply code:

```
538 PBSZ too small. [PBSZmin=numBytes]
 ;  This reply to PBSZ indicates the proposed buffer size is
 ;  unacceptably small. The server may also include an optional
 ;  string of the form PBSZmin=numBytes to indicate the smallest
 ;  buffer size it will accept. The square brackets are not to be
 ;  included in the reply, but indicate that the PBSZmin argument
 ;  in the reply is optional.
```

Protocol Mechanism Naming Specification

The guidelines provided by RFC 2228 for the naming of new protocol mechanisms are somewhat vague, and as such are insufficient for the purposes of successful protocol negotiation as described in section 2. Following is a revised, more rigorous specification for the naming of protocol mechanisms:

The protocol mechanism name must not exceed 255 characters, and may only contain the characters with codes 33...126 in the USASCII character set. Names are case-insensitive. All names must be registered with the IANA, except names beginning with "X-", which are reserved for local use.

4. Miscellaneous changes to RFC 2228

Port 20 restriction for outgoing server data connections:

RFC 2228 is a superset specification of RFC 959 [2], which defines the basic FTP protocol. RFC 959 requires that while in active mode, all outgoing data connections from the FTP server must be bound to port 20, as a weak form of authentication. However, most FTP clients fail to enforce this port restriction (i.e. they don't check). However, implementation experience has shown that the port restriction is also the only hindrance to running a legitimate RFC 2228 server as an entirely user-level proxy (which augments overall system security). Because RFC 2228 provides more comprehensive support for strong, cryptographic authentication mechanisms, it has made the port 20 restriction for outgoing server data connections obsolete, and the restriction is hereby lifted.

5. Security Considerations

This entire memo is about security mechanisms.

6. References

[1] - M. Horowitz and S. J. Lunt. FTP Security Extensions. RFC 2228, October, 1997

[2] - J. Postel and J. Reynolds. File Transfer Protocol (FTP). RFC 959, October, 1985

7. Authors' Addresses

Dan Bonachea
Computer Science Division
566 Soda Hall
Berkeley, CA 94720 USA
Phone: +1 510 642-8493
Email: bonachea@cs.berkeley.edu

Scott McPeak
Computer Science Division
566 Soda Hall
Berkeley, CA 94720 USA
Phone: +1 510 642-8493
Email: smcpeak@cs.berkeley.edu