Network Working Group
Request for Comments: 2595
Category: Standards Track

C. Newman
Innosoft

June 1999

# Using TLS with IMAP, POP3 and ACAP

## Status of this Memo



## Copyright Notice



## 1. Motivation

The TLS protocol (formerly known as SSL) provides a way to secure an application protocol from tampering and eavesdropping. The option of using such security is desirable for IMAP, POP and ACAP due to common connection eavesdropping and hijacking attacks [AUTH]. Although advanced SASL authentication mechanisms can provide a lightweight version of this service, TLS is complimentary to simple authentication-only SASL mechanisms or deployed clear-text password login commands.

Many sites have a high investment in authentication infrastructure (e.g., a large database of a one-way-function applied to user passwords), so a privacy layer which is not tightly bound to user authentication can protect against network eavesdropping attacks without requiring a new authentication infrastructure and/or forcing all users to change their password. Recognizing that such sites will desire simple password authentication in combination with TLS encryption, this specification defines the PLAIN SASL mechanism for use with protocols which lack a simple password authentication command such as ACAP and SMTP. (Note there is a separate RFC for the STARTTLS command in SMTP [SMTPTLS].)

There is a strong desire in the IETF to eliminate the transmission of clear-text passwords over unencrypted channels. While SASL can be used for this purpose, TLS provides an additional tool with different deployability characteristics. A server supporting both TLS with

simple passwords and a challenge/response SASL mechanism is likely to interoperate with a wide variety of clients without resorting to unencrypted clear-text passwords.

The STARTTLS command rectifies a number of the problems with using a separate port for a "secure" protocol variant. Some of these are mentioned in section 7.

### 1.1. Conventions Used in this Document

The key words "REQUIRED", "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [KEYWORDS].

Terms related to authentication are defined in "On Internet Authentication" [AUTH].

Formal syntax is defined using ABNF [ABNF].

In examples, "C:" and "S:" indicate lines sent by the client and server respectively.

## 2. Basic Interoperability and Security Requirements

The following requirements apply to all implementations of the STARTTLS extension for IMAP, POP3 and ACAP.

### 2.1. Cipher Suite Requirements

Implementation of the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA [TLS] cipher suite is REQUIRED. This is important as it assures that any two compliant implementations can be configured to interoperate.

All other cipher suites are OPTIONAL.

### 2.2. Privacy Operational Mode Security Requirements

Both clients and servers SHOULD have a privacy operational mode which refuses authentication unless successful activation of an encryption layer (such as that provided by TLS) occurs prior to or at the time of authentication and which will terminate the connection if that encryption layer is deactivated. Implementations are encouraged to have flexability with respect to the minimal encryption strength or cipher suites permitted. A minimalist approach to this recommendation would be an operational mode where the TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA cipher suite is mandatory prior to permitting authentication.

Clients MAY have an operational mode which uses encryption only when it is advertised by the server, but authentication continues regardless. For backwards compatibility, servers SHOULD have an operational mode where only the authentication mechanisms required by the relevant base protocol specification are needed to successfully authenticate.

## 2.3. Clear-Text Password Requirements

Clients and servers which implement STARTTLS MUST be configurable to refuse all clear-text login commands or mechanisms (including both standards-track and nonstandard mechanisms) unless an encryption layer of adequate strength is active. Servers which allow unencrypted clear-text logins SHOULD be configurable to refuse clear-text logins both for the entire server, and on a per-user basis.

## 2.4. Server Identity Check

During the TLS negotiation, the client MUST check its understanding of the server hostname against the server's identity as presented in the server Certificate message, in order to prevent man-in-the-middle attacks. Matching is performed according to these rules:

- The client MUST use the server hostname it used to open the connection as the value to compare against the server name as expressed in the server certificate. The client MUST NOT use any form of the server hostname derived from an insecure remote source (e.g., insecure DNS lookup). CNAME canonicalization is not done.

- If a subjectAltName extension of type dNSName is present in the certificate, it SHOULD be used as the source of the server's identity.

- Matching is case-insensitive.

- A "*" wildcard character MAY be used as the left-most name component in the certificate. For example, *.example.com would match a.example.com, foo.example.com, etc. but would not match example.com.

- If the certificate contains multiple names (e.g. more than one dNSName field), then a match with any one of the fields is considered acceptable.

If the match fails, the client SHOULD either ask for explicit user confirmation, or terminate the connection and indicate the server's identity is suspect.

### 2.5. TLS Security Policy Check

Both the client and server MUST check the result of the STARTTLS command and subsequent TLS negotiation to see whether acceptable authentication or privacy was achieved. Ignoring this step completely invalidates using TLS for security. The decision about whether acceptable authentication or privacy was achieved is made locally, is implementation-dependent, and is beyond the scope of this document.

## 3. IMAP STARTTLS extension

When the TLS extension is present in IMAP, "STARTTLS" is listed as a capability in response to the CAPABILITY command. This extension adds a single command, "STARTTLS" to the IMAP protocol which is used to begin a TLS negotiation.

### 3.1. STARTTLS Command

Arguments: none

Responses: no specific responses for this command

Result: OK - begin TLS negotiation
BAD - command unknown or arguments invalid

A TLS negotiation begins immediately after the CRLF at the end of the tagged OK response from the server. Once a client issues a STARTTLS command, it MUST NOT issue further commands until a server response is seen and the TLS negotiation is complete.

The STARTTLS command is only valid in non-authenticated state. The server remains in non-authenticated state, even if client credentials are supplied during the TLS negotiation. The SASL [SASL] EXTERNAL mechanism MAY be used to authenticate once TLS client credentials are successfully exchanged, but servers supporting the STARTTLS command are not required to support the EXTERNAL mechanism.

Once TLS has been started, the client MUST discard cached information about server capabilities and SHOULD re-issue the CAPABILITY command. This is necessary to protect against man-in-the-middle attacks which alter the capabilities list prior to STARTTLS. The server MAY advertise different capabilities after STARTTLS.

The formal syntax for IMAP is amended as follows:

```
      command_any   =/  "STARTTLS"

   Example:    C: a001 CAPABILITY
               S: * CAPABILITY IMAP4rev1 STARTTLS LOGINDISABLED
               S: a001 OK CAPABILITY completed
               C: a002 STARTTLS
               S: a002 OK Begin TLS negotiation now
               <TLS negotiation, further commands are under TLS layer>
               C: a003 CAPABILITY
               S: * CAPABILITY IMAP4rev1 AUTH=EXTERNAL
               S: a003 OK CAPABILITY completed
               C: a004 LOGIN joe password
               S: a004 OK LOGIN completed
```

### 3.2. IMAP LOGINDISABLED capability

The current IMAP protocol specification (RFC 2060) requires the implementation of the LOGIN command which uses clear-text passwords. Many sites may choose to disable this command unless encryption is active for security reasons. An IMAP server MAY advertise that the LOGIN command is disabled by including the LOGINDISABLED capability in the capability response. Such a server will respond with a tagged "NO" response to any attempt to use the LOGIN command.

An IMAP server which implements STARTTLS MUST implement support for the LOGINDISABLED capability on unencrypted connections.

An IMAP client which complies with this specification MUST NOT issue the LOGIN command if this capability is present.

This capability is useful to prevent clients compliant with this specification from sending an unencrypted password in an environment subject to passive attacks. It has no impact on an environment subject to active attacks as a man-in-the-middle attacker can remove this capability. Therefore this does not relieve clients of the need to follow the privacy mode recommendation in section 2.2.

Servers advertising this capability will fail to interoperate with many existing compliant IMAP clients and will be unable to prevent those clients from disclosing the user’s password.

## 4. POP3 STARTTLS extension

The POP3 STARTTLS extension adds the STLS command to POP3 servers. If this is implemented, the POP3 extension mechanism [POP3EXT] MUST also be implemented to avoid the need for client probing of multiple commands. The capability name "STLS" indicates this command is present and permitted in the current state.

STLS

Arguments: none

Restrictions:
Only permitted in AUTHORIZATION state.

Discussion:
A TLS negotiation begins immediately after the CRLF at the end of the +OK response from the server. A -ERR response MAY result if a security layer is already active. Once a client issues a STLS command, it MUST NOT issue further commands until a server response is seen and the TLS negotiation is complete.

The STLS command is only permitted in AUTHORIZATION state and the server remains in AUTHORIZATION state, even if client credentials are supplied during the TLS negotiation. The AUTH command [POP-AUTH] with the EXTERNAL mechanism [SASL] MAY be used to authenticate once TLS client credentials are successfully exchanged, but servers supporting the STLS command are not required to support the EXTERNAL mechanism.

Once TLS has been started, the client MUST discard cached information about server capabilities and SHOULD re-issue the CAPA command. This is necessary to protect against man-in-the-middle attacks which alter the capabilities list prior to STLS. The server MAY advertise different capabilities after STLS.

Possible Responses:
+OK -ERR

Examples:
C: STLS
S: +OK Begin TLS negotiation
<TLS negotiation, further commands are under TLS layer>
...
C: STLS
S: -ERR Command not permitted when TLS active

## 5. ACAP STARTTLS extension

When the TLS extension is present in ACAP, "STARTTLS" is listed as a capability in the ACAP greeting. No arguments to this capability are defined at this time. This extension adds a single command, "STARTTLS" to the ACAP protocol which is used to begin a TLS negotiation.

### 5.1. STARTTLS Command

Arguments: none

Responses: no specific responses for this command

Result: OK - begin TLS negotiation
BAD - command unknown or arguments invalid

A TLS negotiation begins immediately after the CRLF at the end of the tagged OK response from the server. Once a client issues a STARTTLS command, it MUST NOT issue further commands until a server response is seen and the TLS negotiation is complete.

The STARTTLS command is only valid in non-authenticated state. The server remains in non-authenticated state, even if client credentials are supplied during the TLS negotiation. The SASL [SASL] EXTERNAL mechanism MAY be used to authenticate once TLS client credentials are successfully exchanged, but servers supporting the STARTTLS command are not required to support the EXTERNAL mechanism.

After the TLS layer is established, the server MUST re-issue an untagged ACAP greeting. This is necessary to protect against man-in-the-middle attacks which alter the capabilities list prior to STARTTLS. The client MUST discard cached capability information and replace it with the information from the new ACAP greeting. The server MAY advertise different capabilities after STARTTLS.

The formal syntax for ACAP is amended as follows:

```
command_any   =/  "STARTTLS"
```

Example:
```
S: * ACAP (SASL "CRAM-MD5") (STARTTLS)
C: a002 STARTTLS
S: a002 OK "Begin TLS negotiation now"
<TLS negotiation, further commands are under TLS layer>
S: * ACAP (SASL "CRAM-MD5" "PLAIN" "EXTERNAL")
```

## 6. PLAIN SASL mechanism

Clear-text passwords are simple, interoperate with almost all existing operating system authentication databases, and are useful for a smooth transition to a more secure password-based authentication mechanism. The drawback is that they are unacceptable for use over an unencrypted network connection.

This defines the "PLAIN" SASL mechanism for use with ACAP and other protocols with no clear-text login command. The PLAIN SASL mechanism MUST NOT be advertised or used unless a strong encryption layer (such as the provided by TLS) is active or backwards compatibility dictates otherwise.

The mechanism consists of a single message from the client to the server. The client sends the authorization identity (identity to login as), followed by a US-ASCII NUL character, followed by the authentication identity (identity whose password will be used), followed by a US-ASCII NUL character, followed by the clear-text password. The client may leave the authorization identity empty to indicate that it is the same as the authentication identity.

The server will verify the authentication identity and password with the system authentication database and verify that the authentication credentials permit the client to login as the authorization identity. If both steps succeed, the user is logged in.

The server MAY also use the password to initialize any new authentication database, such as one suitable for CRAM-MD5 [CRAM-MD5].

Non-US-ASCII characters are permitted as long as they are represented in UTF-8 [UTF-8]. Use of non-visible characters or characters which a user may be unable to enter on some keyboards is discouraged.

The formal grammar for the client message using Augmented BNF [ABNF] follows.

```
message         = [authorize-id] NUL authenticate-id NUL password
authenticate-id = 1*UTF8-SAFE      ; MUST accept up to 255 octets
authorize-id    = 1*UTF8-SAFE      ; MUST accept up to 255 octets
password        = 1*UTF8-SAFE      ; MUST accept up to 255 octets
NUL             = %x00
UTF8-SAFE       = %x01-09 / %x0B-0C / %x0E-7F / UTF8-2 /
                  UTF8-3 / UTF8-4 / UTF8-5 / UTF8-6
UTF8-1          = %x80-BF
UTF8-2          = %xC0-DF UTF8-1
UTF8-3          = %xE0-EF 2UTF8-1
```

```
UTF8-4          = %xF0-F7 3UTF8-1
UTF8-5          = %xF8-FB 4UTF8-1
UTF8-6          = %xFC-FD 5UTF8-1
```

Here is an example of how this might be used to initialize a CRAM-MD5 authentication database for ACAP:

```
Example:    S: * ACAP (SASL "CRAM-MD5") (STARTTLS)
            C: a001 AUTHENTICATE "CRAM-MD5"
            S: + "<1896.697170952@postoffice.reston.mci.net>"
            C: "tim b913a602c7eda7a495b4e6e7334d3890"
            S: a001 NO (TRANSITION-NEEDED)
               "Please change your password, or use TLS to login"
            C: a002 STARTTLS
            S: a002 OK "Begin TLS negotiation now"
            <TLS negotiation, further commands are under TLS layer>
            S: * ACAP (SASL "CRAM-MD5" "PLAIN" "EXTERNAL")
            C: a003 AUTHENTICATE "PLAIN" {21+}
            C: <NUL>tim<NUL>tanstaaftanstaaf
            S: a003 OK CRAM-MD5 password initialized
```

Note: In this example, <NUL> represents a single ASCII NUL octet.

## 7. imaps and pop3s ports

Separate "imaps" and "pop3s" ports were registered for use with SSL. Use of these ports is discouraged in favor of the STARTTLS or STLS commands.

A number of problems have been observed with separate ports for "secure" variants of protocols. This is an attempt to enumerate some of those problems.

- Separate ports lead to a separate URL scheme which intrudes into the user interface in inappropriate ways. For example, many web pages use language like "click here if your browser supports SSL." This is a decision the browser is often more capable of making than the user.

- Separate ports imply a model of either "secure" or "not secure." This can be misleading in a number of ways. First, the "secure" port may not in fact be acceptably secure as an export-crippled cipher suite might be in use. This can mislead users into a false sense of security. Second, the normal port might in fact be secured by using a SASL mechanism which includes a security layer. Thus the separate port distinction makes the complex topic of security policy even more confusing. One common result of this confusion is that firewall administrators are often misled into

permitting the "secure" port and blocking the standard port. This could be a poor choice given the common use of SSL with a 40-bit key encryption layer and plain-text password authentication is less secure than strong SASL mechanisms such as GSSAPI with Kerberos 5.

- Use of separate ports for SSL has caused clients to implement only two security policies: use SSL or don't use SSL. The desirable security policy "use TLS when available" would be cumbersome with the separate port model, but is simple with STARTTLS.

- Port numbers are a limited resource. While they are not yet in short supply, it is unwise to set a precedent that could double (or worse) the speed of their consumption.

## 8. IANA Considerations

This constitutes registration of the "STARTTLS" and "LOGINDISABLED" IMAP capabilities as required by section 7.2.1 of RFC 2060 [IMAP].

The registration for the POP3 "STLS" capability follows:

```
CAPA tag:                   STLS
Arguments:                  none
Added commands:             STLS
Standard commands affected: May enable USER/PASS as a side-effect.
  CAPA command SHOULD be re-issued after successful completion.
Announced states/Valid states: AUTHORIZATION state only.
Specification reference:    this memo
```

The registration for the ACAP "STARTTLS" capability follows:

```
Capability name:            STARTTLS
Capability keyword:         STARTTLS
Capability arguments:       none
Published Specification(s): this memo
Person and email address for further information:
    see author's address section below
```

The registration for the PLAIN SASL mechanism follows:

```
SASL mechanism name:        PLAIN
Security Considerations:    See section 9 of this memo
Published specification:    this memo
Person & email address to contact for further information:
    see author's address section below
Intended usage:             COMMON
Author/Change controller:   see author's address section below
```

## 9. Security Considerations

TLS only provides protection for data sent over a network connection. Messages transferred over IMAP or POP3 are still available to server administrators and usually subject to eavesdropping, tampering and forgery when transmitted through SMTP or NNTP. TLS is no substitute for an end-to-end message security mechanism using MIME security multiparts [MIME-SEC].

A man-in-the-middle attacker can remove STARTTLS from the capability list or generate a failure response to the STARTTLS command. In order to detect such an attack, clients SHOULD warn the user when session privacy is not active and/or be configurable to refuse to proceed without an acceptable level of security.

A man-in-the-middle attacker can always cause a down-negotiation to the weakest authentication mechanism or cipher suite available. For this reason, implementations SHOULD be configurable to refuse weak mechanisms or cipher suites.

Any protocol interactions prior to the TLS handshake are performed in the clear and can be modified by a man-in-the-middle attacker. For this reason, clients MUST discard cached information about server capabilities advertised prior to the start of the TLS handshake.

Clients are encouraged to clearly indicate when the level of encryption active is known to be vulnerable to attack using modern hardware (such as encryption keys with 56 bits of entropy or less).

The LOGINDISABLED IMAP capability (discussed in section 3.2) only reduces the potential for passive attacks, it provides no protection against active attacks. The responsibility remains with the client to avoid sending a password over a vulnerable channel.

The PLAIN mechanism relies on the TLS encryption layer for security. When used without TLS, it is vulnerable to a common network eavesdropping attack. Therefore PLAIN MUST NOT be advertised or used unless a suitable TLS encryption layer is active or backwards compatibility dictates otherwise.

When the PLAIN mechanism is used, the server gains the ability to impersonate the user to all services with the same password regardless of any encryption provided by TLS or other network privacy mechanisms. While many other authentication mechanisms have similar weaknesses, stronger SASL mechanisms such as Kerberos address this issue. Clients are encouraged to have an operational mode where all mechanisms which are likely to reveal the user’s password to the server are disabled.

The security considerations for TLS apply to STARTTLS and the security considerations for SASL apply to the PLAIN mechanism. Additional security requirements are discussed in section 2.

## 10. References

[ABNF] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[ACAP] Newman, C. and J. Myers, "ACAP -- Application Configuration Access Protocol", RFC 2244, November 1997.

[AUTH] Haller, N. and R. Atkinson, "On Internet Authentication", RFC 1704, October 1994.

[CRAM-MD5] Klensin, J., Catoe, R. and P. Krumviede, "IMAP/POP AUTHorize Extension for Simple Challenge/Response", RFC 2195, September 1997.

[IMAP] Crispin, M., "Internet Message Access Protocol - Version 4rev1", RFC 2060, December 1996.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[MIME-SEC] Galvin, J., Murphy, S., Crocker, S. and N. Freed, "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

[POP3] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, May 1996.

[POP3EXT] Gellens, R., Newman, C. and L. Lundblade, "POP3 Extension Mechanism", RFC 2449, November 1998.

[POP-AUTH] Myers, J., "POP3 AUTHentication command", RFC 1734, December 1994.

[SASL] Myers, J., "Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

[SMTPTLS] Hoffman, P., "SMTP Service Extension for Secure SMTP over TLS", RFC 2487, January 1999.

[TLS] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[UTF-8] Yergeau, F., "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

## 11. Author's Address

Chris Newman
Innosoft International, Inc.
1050 Lakes Drive
West Covina, CA 91790 USA

EMail: chris.newman@innosoft.com

## A. Appendix -- Compliance Checklist

An implementation is not compliant if it fails to satisfy one or more of the MUST requirements for the protocols it implements. An implementation that satisfies all the MUST and all the SHOULD requirements for its protocols is said to be "unconditionally compliant"; one that satisfies all the MUST requirements but not all the SHOULD requirements for its protocols is said to be "conditionally compliant".

| Rules | Section |
|---|---|
| Mandatory-to-implement Cipher Suite | 2.1 |
| SHOULD have mode where encryption required | 2.2 |
| server SHOULD have mode where TLS not required | 2.2 |
| MUST be configurable to refuse all clear-text login commands or mechanisms | 2.3 |
| server SHOULD be configurable to refuse clear-text login commands on entire server and on per-user basis | 2.3 |
| client MUST check server identity | 2.4 |
| client MUST use hostname used to open connection | 2.4 |
| client MUST NOT use hostname from insecure remote lookup | 2.4 |
| client SHOULD support subjectAltName of dNSName type | 2.4 |
| client SHOULD ask for confirmation or terminate on fail | 2.4 |
| MUST check result of STARTTLS for acceptable privacy | 2.5 |
| client MUST NOT issue commands after STARTTLS until server response and negotiation done | 3.1,4,5.1 |
| client MUST discard cached information | 3.1,4,5.1,9 |
| client SHOULD re-issue CAPABILITY/CAPA command | 3.1,4 |
| IMAP server with STARTTLS MUST implement LOGINDISABLED | 3.2 |
| IMAP client MUST NOT issue LOGIN if LOGINDISABLED | 3.2 |
| POP server MUST implement POP3 extensions | 4 |
| ACAP server MUST re-issue ACAP greeting | 5.1 |

client SHOULD warn when session privacy not active and/or refuse to proceed without acceptable security level 9
SHOULD be configurable to refuse weak mechanisms or cipher suites 9

The PLAIN mechanism is an optional part of this specification. However if it is implemented the following rules apply:

| Rules | Section |
|---|---|
| MUST NOT use PLAIN unless strong encryption active or backwards compatibility dictates otherwise | 6,9 |
| MUST use UTF-8 encoding for characters in PLAIN | 6 |

## Full Copyright Statement



## Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.