Network Working Group
Request for Comments: 3207
Obsoletes: 2487
Category: Standards Track

P. Hoffman
Internet Mail Consortium

February 2002

SMTP Service Extension for Secure SMTP over Transport Layer Security

Status of this Memo

Copyright Notice

Abstract

This document describes an extension to the SMTP (Simple Mail Transfer Protocol) service that allows an SMTP server and client to use TLS (Transport Layer Security) to provide private, authenticated communication over the Internet. This gives SMTP agents the ability to protect some or all of their communications from eavesdroppers and attackers.

1. Introduction

SMTP [RFC2821] servers and clients normally communicate in the clear over the Internet. In many cases, this communication goes through one or more router that is not controlled or trusted by either entity. Such an untrusted router might allow a third party to monitor or alter the communications between the server and client.

Further, there is often a desire for two SMTP agents to be able to authenticate each others' identities. For example, a secure SMTP server might only allow communications from other SMTP agents it knows, or it might act differently for messages received from an agent it knows than from one it doesn't know.

TLS [TLS], more commonly known as SSL, is a popular mechanism for enhancing TCP communications with privacy and authentication. TLS is in wide use with the HTTP protocol, and is also being used for adding security to many other common protocols that run over TCP.

This document obsoletes RFC 2487.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. STARTTLS Extension

The STARTTLS extension to SMTP is laid out as follows:

(1) the name of the SMTP service defined here is STARTTLS;

(2) the EHLO keyword value associated with the extension is STARTTLS;

(3) the STARTTLS keyword has no parameters;

(4) a new SMTP verb, "STARTTLS", is defined;

(5) no additional parameters are added to any SMTP command.

3. The STARTTLS Keyword

The STARTTLS keyword is used to tell the SMTP client that the SMTP server is currently able to negotiate the use of TLS. It takes no parameters.

4. The STARTTLS Command

The format for the STARTTLS command is:

STARTTLS

with no parameters.

After the client gives the STARTTLS command, the server responds with one of the following reply codes:

220 Ready to start TLS
501 Syntax error (no parameters allowed)
454 TLS not available due to temporary reason

If the client receives the 454 response, the client must decide whether or not to continue the SMTP session. Such a decision is based on local policy. For instance, if TLS was being used for client authentication, the client might try to continue the session, in case the server allows it even with no authentication. However, if TLS was being negotiated for encryption, a client that gets a 454 response needs to decide whether to send the message anyway with no TLS encryption, whether to wait and try again later, or whether to give up and notify the sender of the error.

A publicly-referenced SMTP server MUST NOT require use of the STARTTLS extension in order to deliver mail locally. This rule prevents the STARTTLS extension from damaging the interoperability of the Internet's SMTP infrastructure. A publicly-referenced SMTP server is an SMTP server which runs on port 25 of an Internet host listed in the MX record (or A record if an MX record is not present) for the domain name on the right hand side of an Internet mail address.

Any SMTP server may refuse to accept messages for relay based on authentication supplied during the TLS negotiation. An SMTP server that is not publicly referenced may refuse to accept any messages for relay or local delivery based on authentication supplied during the TLS negotiation.

A SMTP server that is not publicly referenced may choose to require that the client perform a TLS negotiation before accepting any commands. In this case, the server SHOULD return the reply code:

530 Must issue a STARTTLS command first

to every command other than NOOP, EHLO, STARTTLS, or QUIT. If the client and server are using the ENHANCEDSTATUSCODES ESMTP extension [RFC2034], the status code to be returned SHOULD be 5.7.0.

After receiving a 220 response to a STARTTLS command, the client MUST start the TLS negotiation before giving any other SMTP commands. If, after having issued the STARTTLS command, the client finds out that some failure prevents it from actually starting a TLS handshake, then it SHOULD abort the connection.

If the SMTP client is using pipelining as defined in RFC 2920, the STARTTLS command must be the last command in a group.

4.1 Processing After the STARTTLS Command

After the TLS handshake has been completed, both parties MUST immediately decide whether or not to continue based on the authentication and privacy achieved. The SMTP client and server may decide to move ahead even if the TLS negotiation ended with no authentication and/or no privacy because most SMTP services are performed with no authentication and no privacy, but some SMTP clients or servers may want to continue only if a particular level of authentication and/or privacy was achieved.

If the SMTP client decides that the level of authentication or privacy is not high enough for it to continue, it SHOULD issue an SMTP QUIT command immediately after the TLS negotiation is complete. If the SMTP server decides that the level of authentication or privacy is not high enough for it to continue, it SHOULD reply to every SMTP command from the client (other than a QUIT command) with the 554 reply code (with a possible text string such as "Command refused due to lack of security").

The decision of whether or not to believe the authenticity of the other party in a TLS negotiation is a local matter. However, some general rules for the decisions are:

- A SMTP client would probably only want to authenticate an SMTP server whose server certificate has a domain name that is the domain name that the client thought it was connecting to.
- A publicly-referenced SMTP server would probably want to accept any verifiable certificate from an SMTP client, and would possibly want to put distinguishing information about the certificate in the Received header of messages that were relayed or submitted from the client.

4.2 Result of the STARTTLS Command

Upon completion of the TLS handshake, the SMTP protocol is reset to the initial state (the state in SMTP after a server issues a 220 service ready greeting). The server MUST discard any knowledge obtained from the client, such as the argument to the EHLO command, which was not obtained from the TLS negotiation itself. The client MUST discard any knowledge obtained from the server, such as the list of SMTP service extensions, which was not obtained from the TLS negotiation itself. The client SHOULD send an EHLO command as the first command after a successful TLS negotiation.

The list of SMTP service extensions returned in response to an EHLO command received after the TLS handshake MAY be different than the list returned before the TLS handshake. For example, an SMTP server

might not want to advertise support for a particular SASL mechanism [SASL] unless a client has sent an appropriate client certificate during a TLS handshake.

Both the client and the server MUST know if there is a TLS session active. A client MUST NOT attempt to start a TLS session if a TLS session is already active. A server MUST NOT return the STARTTLS extension in response to an EHLO command received after a TLS handshake has completed.

4.3 STARTTLS on the Submission Port

STARTTLS is a valid ESMTP extension when used on the Submission port, as defined in [RFC2476]. In fact, since the submission port is by definition not a publicly referenced SMTP server, the STARTTLS extension can be particularly useful by providing security and authentication for this service.

5. Usage Example

The following dialog illustrates how a client and server can start a TLS session:

```
S: <waits for connection on TCP port 25>
C: <opens connection>
S: 220 mail.imc.org SMTP service ready
C: EHLO mail.example.com
S: 250-mail.imc.org offers a warm hug of welcome
S: 250-8BITMIME
S: 250-STARTTLS
S: 250 DSN
C: STARTTLS
S: 220 Go ahead
C: <starts TLS negotiation>
C & S: <negotiate a TLS session>
C & S: <check result of negotiation>
C: EHLO mail.example.com
S: 250-mail.imc.org touches your hand gently for a moment
S: 250-8BITMIME
S: 250 DSN
```

6. Security Considerations

It should be noted that SMTP is not an end-to-end mechanism. Thus, if an SMTP client/server pair decide to add TLS privacy, they are not securing the transport from the originating mail user agent to the recipient. Further, because delivery of a single piece of mail may go between more than two SMTP servers, adding TLS privacy to one pair

of servers does not mean that the entire SMTP chain has been made private. Further, just because an SMTP server can authenticate an SMTP client, it does not mean that the mail from the SMTP client was authenticated by the SMTP client when the client received it.

Both the SMTP client and server must check the result of the TLS negotiation to see whether an acceptable degree of authentication and 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.

The SMTP client and server should note carefully the result of the TLS negotiation. If the negotiation results in no privacy, or if it results in privacy using algorithms or key lengths that are deemed not strong enough, or if the authentication is not good enough for either party, the client may choose to end the SMTP session with an immediate QUIT command, or the server may choose to not accept any more SMTP commands.

A man-in-the-middle attack can be launched by deleting the "250 STARTTLS" response from the server. This would cause the client not to try to start a TLS session. Another man-in-the-middle attack is to allow the server to announce its STARTTLS capability, but to alter the client's request to start TLS and the server's response. In order to defend against such attacks both clients and servers MUST be able to be configured to require successful TLS negotiation of an appropriate cipher suite for selected hosts before messages can be successfully transferred. The additional option of using TLS when possible SHOULD also be provided. An implementation MAY provide the ability to record that TLS was used in communicating with a given peer and generating a warning if it is not used in a later session.

If the TLS negotiation fails or if the client receives a 454 response, the client has to decide what to do next. There are three main choices: go ahead with the rest of the SMTP session, retry TLS at a later time, or give up and return the mail to the sender. If a failure or error occurs, the client can assume that the server may be able to negotiate TLS in the future, and should try negotiate TLS in a later session, until some locally-chosen timeout occurs, at which point, the client should return the mail to the sender. However, if the client and server were only using TLS for authentication, the client may want to proceed with the SMTP session, in case some of the operations the client wanted to perform are accepted by the server even if the client is unauthenticated.

Before the TLS handshake has begun, any protocol interactions are performed in the clear and may be modified by an active attacker.

For this reason, clients and servers MUST discard any knowledge obtained prior to the start of the TLS handshake upon completion of the TLS handshake.

The STARTTLS extension is not suitable for authenticating the author of an email message unless every hop in the delivery chain, including the submission to the first SMTP server, is authenticated. Another proposal [SMTP-AUTH] can be used to authenticate delivery and MIME security multiparts [MIME-SEC] can be used to authenticate the author of an email message. In addition, the [SMTP-AUTH] proposal offers simpler and more flexible options to authenticate an SMTP client and the SASL EXTERNAL mechanism [SASL] MAY be used in conjunction with the STARTTLS command to provide an authorization identity.

7. References

[RFC2821] Klensin, J., "Simple Mail Transfer Protocol", RFC 2821, April 2001.

[RFC2034] Freed, N., "SMTP Service Extension for Returning Enhanced Error Codes", RFC 2034, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2476] Gellens, R. and J. Klensin, "Message Submission", RFC 2476, December 1998.

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

[SMTP-AUTH] Myers, J., "SMTP Service Extension for Authentication", RFC 2554, March 1999.

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

Appendix

This document is a revision of RFC 2487, which is a Proposed Standard. The changes from that document are:

- Section 5 and 7: More discussion of the man-in-the-middle attacks
- Section 5: Additional discussion of when a server should and should not advertise the STARTTLS extension
- Section 5: Changed the requirements on SMTP clients after receiving a 220 response.
- Section 5.1: Clarified description of verifying certificates.
- Section 5.3: Added the section on "STARTTLS on the Submission Port"
- Section 6: Bug fix in the example to indicate that the client needs to issue a new EHLO command, as already is described in section 5.2.
- Section 7: Clarification of the paragraph on acceptable degree of privacy. Significant change to the discussion of how to avoid a man-in-the-middle attack.
- Section A: Update reference from RFC 821 to RFC 2821.

Author's Address

Paul Hoffman
Internet Mail Consortium
127 Segre Place
Santa Cruz, CA 95060

Phone: (831) 426-9827
EMail: phoffman@imc.org

Full Copyright Statement

Acknowledgement

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