Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 17, 2017

J. Fenton
Altmode Networks

February 13, 2017

SMTP Require TLS Option
draft-fenton-smtp-require-tls-03

Abstract

The SMTP STARTTLS option, used in negotiating transport-level encryption of SMTP connections, is not as useful from a security standpoint as it might be because of its opportunistic nature; message delivery is prioritized over security. This document describes a complementary SMTP service extension, REQUIRETLS. If the REQUIRETLS option is used when sending a message, it asserts a request on the part of the message sender to override the default negotiation of TLS, either by requiring that TLS be negotiated when the message is relayed, or by requesting that policy mechanisms such as SMTP STS and DANE be ignored when relaying a high priority message.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 17, 2017.

Copyright Notice

Table of Contents

1. Introduction 2
1.1. Requirements Language 3
2. The REQUIRETLS Service Extension 3
3. REQUIRETLS Semantics 5
3.1. REQUIRETLS Receipt Requirements 5
3.2. REQUIRETLS Sender Requirements 5
3.2.1. Sending with TLS Required 5
3.2.2. Sending with TLS Optional 6
3.3. REQUIRETLS Submission 7
3.4. Delivery of REQUIRETLS messages 7
4. Non-delivery message handling 7
5. Mailing list considerations 8
6. IANA Considerations 8
7. Security Considerations 8
7.1. Passive attacks 9
7.2. Active attacks 9
7.3. Bad Actor MTAs 9
8. Acknowledgements 10
9. Revision History 10
9.1. Changes Since -02 Draft 10
9.2. Changes Since -01 Draft 10
9.3. Changes Since -00 Draft 10
10. References 11
10.1. Normative References 11
10.2. Informative References 12
Author’s Address 13

1. Introduction

The SMTP [RFC5321] STARTTLS service extension [RFC3207] provides a means by which an SMTP server and client can establish a Transport Layer Security (TLS) protected session for the transmission of email messages. By default, TLS is used only upon mutual agreement (successful negotiation) of STARTTLS between the client and server; if this is not possible, the message is sent without transport encryption. Furthermore, it is common practice for the client to negotiate TLS even if the SMTP server’s certificate fails to authenticate it.

Policy mechanisms such as DANE [RFC7672] and SMTP STS [I-D.ietf-uta-mta-sts] may impose requirements for the use of TLS for email destined for some domains. However, such policies do not allow the sender to specify which messages are more sensitive and require transport-level encryption, and which ones are urgent and ought to be relayed even if TLS cannot be negotiated successfully.

The default opportunistic nature of SMTP TLS enables several "on the wire" attacks on SMTP security between MTAs. These include passive eavesdropping on connections for which TLS is not used, interference in the SMTP protocol to prevent TLS from being negotiated (presumably followed by eavesdropping), and insertion of a man-in-the-middle attacker taking advantage of the lack of server authentication by the client. Attacks are more described in more detail in the Security Considerations section of this document.

The REQUIRETLS SMTP service extension allows the SMTP client to specify that a given message sent during a particular session MUST be sent over a TLS protected session with specified security characteristics, or conversely that delivery should be prioritized over ability to negotiate TLS. For messages requiring TLS negotiation, it also requires that the SMTP server advertise that it also supports REQUIRETLS, in effect promising that it will honor the requirement to require TLS transmission and REQUIRETLS support for onward transmission of messages specifying that requirement.

1.1. Requirements Language

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 RFC 2119 [RFC2119].

2. The REQUIRETLS Service Extension

1. The textual name of the extension is "Require TLS".

2. The EHLO keyword value associated with this extension is "REQUIRETLS".

3. One MAIL FROM option is defined by this extension.

4. Two new SMTP status codes are defined by this extension to convey error conditions resulting from failure of the client to negotiate a TLS connection with the required security and as a result of an attempt to send to a server not also supporting the REQUIRETLS extension.

In order to specify REQUIRETLS treatment for a given message, the REQUIRETLS option is specified on the MAIL FROM command when that message is transmitted. With the exception of REQUIRETLS=NO (described below), this option MUST only be specified in the context of an SMTP session meeting the security requirements that have been specified:

- The session itself MUST employ TLS transmission, unless the NO parameter is specified.

- Any server authentication requirements specified as an option to the REQUIRETLS option (see below) MUST have been satisfied in establishing the current session.

An optional parameter to the REQUIRETLS MAIL FROM option specifies the requirements for server authentication that MUST be used for any onward transmission of the following message. The parameter takes the form of either a single value or comma-separated list, separated from the REQUIRETLS option by a single "=" (equals-sign) character. If present, the parameter MUST take one or more of the following values:

- CHAIN - The certificate presented by the SMTP server MUST verify successfully in a trust chain leading to a certificate trusted by the SMTP client. The choice of trusted (root) certificates by the client is at their own discretion. The client MAY choose to use the certificate set maintained by the CA/B forum [citation needed] for this purpose.

- DANE - The certificate presented by the SMTP server MUST verify succesfully using DANE as specified in RFC 7672 [RFC7672].

- DNSSEC - The server MUST confirm that any MX record or CNAME lookup used to locate the SMTP server must be DNSSEC [RFC4035] signed and valid.

- NO - The SMTP client SHOULD attempt to send the message regardless of its ability to negotiate STARTTLS with the SMTP server, ignoring policy-based mechanisms, if any, asserted by the recipient domain. Nevertheless, the client MAY negotiate STARTTLS with the server if available. If the NO parameter is present, any other REQUIRETLS parameter MUST NOT be used.

The CHAIN and DANE parameters are additive; if both are specified, either method of certificate validation is acceptable. If neither CHAIN nor DANE is specified, the certificate presented by the SMTP server is not required to be verified.

3. REQUIRETLS Semantics

3.1. REQUIRETLS Receipt Requirements

Upon receipt of the REQUIRETLS option on a MAIL FROM command during the receipt of a message, an SMTP server MUST tag that message as needing REQUIRETLS handling with the specified option(s). The manner in which this tagging takes place is implementation-dependent. If the message is being locally aliased and redistributed to multiple addresses, all instances of the message MUST be tagged in the same manner.

3.2. REQUIRETLS Sender Requirements

3.2.1. Sending with TLS Required

When sending a message tagged with REQUIRETLS other than the REQUIRETLS=NO option, the sending (client) MTA MUST:

1. Look up the SMTP server to which the message is to be sent. If the DNSSEC option is included in the message tag, the MX record lookups in this process MUST use DNSSEC verification and the response(s) MUST be DNSSEC-signed in order to ensure the integrity of the resource identifier [RFC6125] used to authenticate the SMTP server.

2. Open an SMTP session with the peer SMTP server using the EHLO verb. The server MUST advertise the REQUIRETLS capability.

3. Establish a TLS-protected SMTP session with its peer SMTP server and authenticate the server’s certificate with the specified authentication method.

4. The SMTP client SHOULD also require that meaningfully secure cipher algorithms and key lengths be negotiated with the server. The choices of key lengths and algorithms change over time, so a specific requirement is not presented here.

If any of the above steps fail, the client SHOULD issue a QUIT to the server and repeat steps 2-4 with each host on the recipient domain’s list of MX hosts in an attempt to find a mail path that meets the sender’s requirements. If there are no more MX hosts or if the MX record lookup is not DNSSEC-protected and DNSSEC verification is required, the client MUST NOT transmit the message and MUST issue an SMTP QUIT command to the server. The client MAY send other, unprotected, messages to that server prior to issuing the QUIT if it has any.

Following such a failure, the SMTP client MUST send a non-delivery notification to the reverse-path of the failed message as described in section 3.6 of [RFC5321]. The following status codes [RFC5248] SHOULD be used:

- DNSSEC lookup failure: 5.x.x DNSSEC lookup required
- REQUIRETLS not supported by server: 5.7.x REQUIRETLS needed
- Unable to establish TLS-protected SMTP session: 5.7.10 Encryption needed

Refer to Section 4. for further requirements regarding non-delivery messages.

If all REQUIRETLS requirements have been met, transmit the message, issuing the REQUIRETLS option on the MAIL FROM command with the required option(s), if any.

3.2.2. Sending with TLS Optional

Messages tagged REQUIRETLS=NO are handled differently from other REQUIRETLS messages, as follows. When sending a message tagged with REQUIRETLS=NO, the sending (client) MTA MUST:

- Look up the SMTP server to which the message is to be sent as described in [RFC5321].
- Open an SMTP session with the peer SMTP server using the EHLO verb. Attempt to negotiate STARTTLS if possible, and follow any policy published by the recipient domain, but do not fail if this is unsuccessful.
- If the server does not advertise the REQUIRETLS capability, send the message in the usual manner (without the REQUIRETLS option, because the server will not understand the option).
- If the server advertises the REQUIRETLS capability, send the message with the REQUIRETLS=NO option.

Some SMTP servers that are configured to expect STARTTLS connections as a matter of policy may not accept messages in the absence of STARTTLS. This MUST be expected, and a non-delivery notification returned to the sender.

Messages tagged with the REQUIRETLS=NO option will be sent without the option to SMTP servers not supporting REQUIRETLS. REQUIRETLS=NO MAY therefore not persist through multiple email relay hops.

3.3. REQUIRETLS Submission

An MUA or other agent making the initial introduction of a message to SMTP has authority to decide whether to require TLS, and if so, using what authentication method(s). It does so by issuing the REQUIRETLS option in the MAIL FROM command during message submission. This MAY be done based on a user interface selection, on a header field included in the message, or based on policy. The manner in which the decision to require TLS is made is implementation-dependent and is beyond the scope of this specification.

3.4. Delivery of REQUIRETLS messages

Messages are usually retrieved by end users using protocols other than SMTP such as IMAP [RFC3501], POP [RFC1939], or web mail systems. Mail delivery agents supporting REQUIRETLS SHOULD require that retrieval of messages requiring transport encryption take place over authenticated, encrypted channels.

4. Non-delivery message handling

Non-delivery ("bounce") messages usually contain important metadata about the message to which they refer, including the original message header. They therefore MUST be protected in the same manner as the original message. All non-delivery messages, whether resulting from a REQUIRETLS error or some other, MUST employ REQUIRETLS using the same authentication method(s) as the message that caused the error to occur.

It should be noted that the path from the origination of an error bounce message back to the MAIL FROM address may not share the same REQUIRETLS support as the forward path. Therefore, users of REQUIRETLS (other than REQUIRETLS=NO) are advised to make sure that they are capable of receiving mail using REQUIRETLS at the same authentication method(s) as messages they send. Otherwise, such non-delivery messages will be lost.

If unable to send a bounce message due to a REQUIRETLS failure (the return path not supporting the TLS requirements in the original message), the MTA sending the bounce message MAY send a redacted non-delivery message to the postmaster of the domain identified in the envelope-From address identifying the message only by Message-ID and indicating the type of failure. The original From, Return-path, To, Sender, Cc, and related header fields MUST NOT be included in this message.

Senders of messages specifying REQUIRETLS (other than REQUIRETLS=NO) are advised to consider the increased likelihood that bounce messages will be lost as a result of REQUIRETLS return path failure.

5. Mailing list considerations

Mailing lists, upon receipt of a message, originate new messages to list addresses, as distinct from an aliasing operation that redirects the original message, in some cases to multiple recipients. The requirement to preserve the REQUIRETLS tag and options therefore does not extend to mailing lists. REQUIRETLS users SHOULD use caution when sending to mailing lists and MUST NOT assume that REQUIRETLS applies to messages from the list operator to list members.

Mailing list operators MAY apply REQUIRETLS requirements in incoming messages to the resulting messages they originate. If this is done, they SHOULD also apply these requirements to administrative traffic, such as messages to moderators requesting approval of messages.

6. IANA Considerations

If published as an RFC, this draft requests the addition of the keyword REQUIRETLS to the SMTP Service Extensions Registry [MailParams].

If published as an RFC, this draft also requests the creation of a registry, REQUIRETLS Security Requirements, to be initially populated with the CHAIN, DANE, DNSSEC, and NO keywords.

If published as an RFC, this draft requests the addition of an entry to the Simple Mail Transfer Protocol (SMTP) Enhanced Status Codes Registry [SMTPStatusCodes] in the 5.7.YYY range to indicate lack of REQUIRETLS support by an SMTP server to which a message is being routed.

This section is to be removed during conversion into an RFC by the RFC Editor.

7. Security Considerations

The purpose of REQUIRETLS is to improve communications security for email by giving the originator of a message an expectation that it will be transmitted in an encrypted form "over the wire". When used, REQUIRETLS changes the traditional behavior of email transmission, which favors delivery over the ability to send email messages using transport-layer security, to one in which requested security takes precedence over delivery and domain-level policy.

The following considerations apply to STARTTLS other than the STARTTLS=NO option, since messages specifying that option are specifying less concern with transport security.

7.1. Passive attacks

REQUIRETLS is generally effective against passive attackers who are merely trying to eavesdrop on an SMTP exchange between an SMTP client and server. This assumes, of course, the cryptographic integrity of the TLS connection being used.

7.2. Active attacks

Active attacks against TLS encrypted SMTP connections can take many forms. One such attack is to interfere in the negotiation by changing the STARTTLS command to something illegal such as XXXXXXXX. This causes TLS negotiation to fail and messages to be sent in the clear, where they can be intercepted. REQUIRETLS detects the failure of STARTTLS and declines to send the message rather than send it insecurely.

A second form of attack is a man-in-the-middle attack where the attacker terminates the TLS connection rather than the intended SMTP server. This is possible when, as is commonly the case, the SMTP client either does not verify the server's certificate or establishes the connection even when the verification fails. The REQUIRETLS CHAIN and DANE options allow the message sender to specify that successful certificate validation, using either or both of two different methods, is required before sending the message.

Another active attack involves the spoofing of DNS MX records of the recipient domain. An attacker having this capability could cause the message to be redirected to a mail server under the attacker's own control, which would presumably have a valid certificate. The REQUIRETLS DNSSEC option allows the message sender to require that valid DNSSEC [RFC4033] signatures be obtained when locating the recipient's mail server, in order to address that attack.

In addition to support of the DNSSEC option, domains receiving email SHOULD deploy DNSSEC and SMTP clients SHOULD deploy DNSSEC verification.

7.3. Bad Actor MTAs

A bad-actor MTA along the message transmission path could misrepresent its support of REQUIRETLS and/or actively strip REQUIRETLS tags from messages it handles. However, since intermediate MTAs are already trusted with the cleartext of messages

they handle, and are not part of the threat model for transport-layer security, they are also not part of the threat model for REQUIRETLS.

It should be reemphasized that since SMTP TLS is a transport-layer security protocol, messages sent using REQUIRETLS are not encrypted end-to-end and are visible to MTAs that are part of the message delivery path. Messages containing sensitive information that MTAs should not have access to MUST be sent using end-to-end content encryption such as OpenPGP [RFC4880] or S/MIME [RFC5751].

8. Acknowledgements

The author would like to acknowledge many helpful suggestions on the ietf-smtp and uta mailing lists, in particular those of Viktor Dukhovni, Tony Finch, Jeremy Harris, Arvel Hathcock, John Klensin, John Levine, Rolf Sonneveld, and Per Thorsheim.

9. Revision History

To be removed by RFC Editor upon publication as an RFC.

9.1. Changes Since -02 Draft

- Incorporation of "MAY TLS" functionality as REQUIRETLS=NO per suggestion on UTA WG mailing list.
- Additional guidance on bounce messages

9.2. Changes Since -01 Draft

- Specified retries when multiple MX hosts exist for a given domain.
- Clarified generation of non-delivery messages
- Specified requirements for application of REQUIRETLS to mail forwarders and mailing lists.
- Clarified DNSSEC requirements to include MX lookup only.
- Corrected terminology regarding message retrieval vs. delivery.
- Changed category to standards track.

9.3. Changes Since -00 Draft

- Conversion of REQUIRETLS from an SMTP verb to a MAIL FROM parameter to better associate REQUIRETLS requirements with transmission of individual messages.

- Addition of an option to require DNSSEC lookup of the remote mail server, since this affects the common name of the certificate that is presented.

- Clarified the wording to more clearly state that TLS sessions must be established and not simply that STARTTLS is negotiated.

- Introduced need for minimum encryption standards (key lengths and algorithms)

- Substantially rewritten Security Considerations section

10. References

10.1. Normative References

[MailParams] Internet Assigned Numbers Authority (IANA), "IANA Mail Parameters", 2007, <http://www.iana.org/assignments/mail-parameters>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, DOI 10.17487/RFC3207, February 2002, <http://www.rfc-editor.org/info/rfc3207>.

[RFC4035] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Protocol Modifications for the DNS Security Extensions", RFC 4035, DOI 10.17487/RFC4035, March 2005, <http://www.rfc-editor.org/info/rfc4035>.

[RFC5248] Hansen, T. and J. Klensin, "A Registry for SMTP Enhanced Mail System Status Codes", BCP 138, RFC 5248, DOI 10.17487/RFC5248, June 2008, <http://www.rfc-editor.org/info/rfc5248>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <http://www.rfc-editor.org/info/rfc5321>.

[SMTPStatusCodes]
Internet Assigned Numbers Authority (IANA), "Simple Mail Transfer Protocol (SMTP) Enhanced Status Codes Registry", 2008, <http://www.iana.org/assignments/smtp-enhanced-status-codes>.

10.2. Informative References

[I-D.ietf-uta-mta-sts]
Margolis, D., Risher, M., Ramakrishnan, B., Brotman, A., and J. Jones, "SMTP MTA Strict Transport Security (MTA-STS)", draft-ietf-uta-mta-sts-02 (work in progress), December 2016.

[RFC1939] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, DOI 10.17487/RFC1939, May 1996, <http://www.rfc-editor.org/info/rfc1939>.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, DOI 10.17487/RFC3501, March 2003, <http://www.rfc-editor.org/info/rfc3501>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <http://www.rfc-editor.org/info/rfc4880>.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, DOI 10.17487/RFC5751, January 2010, <http://www.rfc-editor.org/info/rfc5751>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <http://www.rfc-editor.org/info/rfc6125>.

[RFC7672] Dukhovni, V. and W. Hardaker, "SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)", RFC 7672, DOI 10.17487/RFC7672, October 2015, <http://www.rfc-editor.org/info/rfc7672>.

Author's Address

Jim Fenton
Altmode Networks
704 Benvenue Avenue
Los Altos, California 94024
USA

Email: fenton@bluepopcorn.net

Network Working Group
Internet-Draft
Updates: 1939, 2595, 3464, 3501, 5068,
6186, 6409 (if approved)
Intended status: Standards Track
Expires: June 9, 2018

K. Moore
Windrock, Inc.
C. Newman
Oracle

December 6, 2017

Cleartext Considered Obsolete: Use of TLS for Email Submission and Access
draft-ietf-uta-email-deep-12

Abstract

This specification outlines current recommendations for the use of Transport Layer Security (TLS) to provide confidentiality of email traffic between a mail user agent (MUA) and a mail submission or mail access server.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 9, 2018.

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions and Terminology Used in This Document 3
3. Implicit TLS 4
3.1. Implicit TLS for POP 5
3.2. Implicit TLS for IMAP 5
3.3. Implicit TLS for SMTP Submission 5
3.4. Implicit TLS Connection Closure for POP, IMAP and SMTP Submission 6
4. Use of TLS by Mail Access Services and Message Submission Services 6
4.1. Deprecation of Services Using Cleartext and TLS Versions < 1.1 8
4.2. Mail Server Use of Client Certificate Authentication . . 9
4.3. Recording TLS Cipher Suite in Received Header 9
4.4. TLS Server Certificate Requirements 10
4.5. Recommended DNS records for mail protocol servers 10
4.5.1. MX records 10
4.5.2. SRV records 10
4.5.3. DNSSEC 10
4.5.4. TLSA records 11
4.6. Changes to Internet Facing Servers 11
5. Use of TLS by Mail User Agents 11
5.1. Use of SRV records in Establishing Configuration 12
5.2. Minimum Confidentiality Level 13
5.3. Certificiate Validation 14
5.4. Certificate Pinning 15
5.5. Client Certificate Authentication 15
6. Considerations related to Anti-Virus/Anti-Spam Software and Services 16
7. IANA Considerations 16
7.1. POP3S Port Registration Update 17
7.2. IMAPS Port Registration Update 17
7.3. Submissions Port Registration 17
7.4. Additional registered clauses for Received fields 18
8. Security Considerations 18
9. References 19
9.1. Normative References 19
9.2. Informative References 21
Appendix A. Design Considerations 22
Appendix B. Change Log 24
Appendix C. Acknowledgements 29
Authors' Addresses 29

1. Introduction

Software that provides email service via Internet Message Access Protocol (IMAP) [RFC3501], Post Office Protocol (POP) [RFC1939] and/or Simple Mail Transfer Protocol (SMTP) Submission [RFC6409] usually has Transport Layer Security (TLS) [RFC5246] support but often does not use it in a way that maximizes end-user confidentiality. This specification describes current recommendations for the use of TLS in interactions between Mail User Agents and Mail Access Services, and between Mail User Agents and Mail Submission Services.

In brief, this memo now recommends that:

- TLS version 1.2 or greater be used for all traffic between mail user agents (MUAs) and mail submission servers, and also between MUAs and mail access servers.

- MUAs and mail service providers discourage use of cleartext protocols for mail access and mail submission, and deprecate use of cleartext protocols for these purposes as soon as practicable.

- Use of "Implicit TLS" on ports reserved for that purpose, in preference to STARTTLS on a port that otherwise supports cleartext.

This memo does not address use of TLS with SMTP for message relay (where Message Submission [RFC6409] does not apply). Improved use of TLS with SMTP for message relay requires a different approach. One approach to address that topic is described in [RFC7672]; another is in [I-D.ietf-uta-mta-sts].

The recommendations in this memo do not replace the functionality of, and are not intended as a substitute for, end-to-end encryption of electronic mail.

2. Conventions and Terminology Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

The term "Implicit TLS" refers to the automatic negotiation of TLS whenever a TCP connection is made on a particular TCP port that is used exclusively by that server for TLS connections. The term "Implicit TLS" is intended to contrast with use of STARTTLS and similar commands in POP, IMAP, SMTP message submission, and other

protocols, that are used by client and server to explicitly negotiate TLS on an established cleartext TCP connection.

The term "Mail Access Services" includes POP, IMAP and any other protocol used to access or modify received messages, or to access or modify a mail user's account configuration.

"Mail Submission Service" refers to the use of the protocol specified in [RFC6409] (or one of its predecessors or successors) for submission of outgoing messages for delivery to recipients.

The term "Mail Service Provider" (MSP) refers to a provider of Mail Access Services and/or Mail Submission Services.

The term "Mail Account" refers to a user's identity with a Mail Service Provider, that user's authentication credentials, any user email that is stored by the MSP, and any other per-user configuration information maintained by the MSP (for example, spam filtering instructions). Most Mail User Agents (MUAs) support the ability to access multiple Mail Accounts.

For each account that an MUA accesses on its user's behalf, it must have the server names, ports, authentication credentials, and other configuration information specified by the user. This information which is used by the MUA is referred to as "Mail Account Configuration"

This specification expresses syntax using the Augmented Backus-Naur Form (ABNF) as described in [RFC5234], including the core rules in Appendix B and rules from [RFC5322].

3. Implicit TLS

Previous standards for use of email protocols with TLS used the STARTTLS mechanism: [RFC2595], [RFC3207], and [RFC3501]. With STARTTLS, the client establishes a cleartext application session and determines whether to issue a STARTTLS command based on server capabilities and client configuration. If the client issues a STARTTLS command, a TLS handshake follows that can upgrade the connection. While this mechanism has been deployed, an alternate mechanism where TLS is negotiated immediately at connection start on a separate port (referred to in this document as "Implicit TLS") has been deployed more successfully. To encourage more widespread use of TLS, and to encourage a greater consistency for how TLS is used, this specification now recommends use of Implicit TLS for POP, IMAP, SMTP Submission, and all other protocols used between a Mail User Agent and a mail service.

3.1. Implicit TLS for POP

When a TCP connection is established for the "pop3s" service (default port 995), a TLS handshake begins immediately. Clients MUST implement the certificate validation mechanism described in [RFC7817]. Once the TLS session is established, POP3 [RFC1939] protocol messages are exchanged as TLS application data for the remainder of the TCP connection. After the server sends a +OK greeting, the server and client MUST enter AUTHORIZATION state, even if a client certificate was supplied during the TLS handshake.

See Section 5.5 and Section 4.2 for additional information on client certificate authentication. See Section 7.1 for port registration information.

3.2. Implicit TLS for IMAP

When a TCP connection is established for the "imaps" service (default port 993), a TLS handshake begins immediately. Clients MUST implement the certificate validation mechanism described in [RFC7817]. Once the TLS session is established, IMAP [RFC3501] protocol messages are exchanged as TLS application data for the remainder of the TCP connection. If a client certificate was provided during the TLS handshake that the server finds acceptable, the server MAY issue a PREAUTH greeting in which case both the server and client enter AUTHENTICATED state. If the server issues an OK greeting then both server and client enter NOT AUTHENTICATED state.

See Section 5.5 and Section 4.2 for additional information on client certificate authentication. See Section 7.1 and Section 7.2 for port registration information.

3.3. Implicit TLS for SMTP Submission

When a TCP connection is established for the "submissions" service (default port 465), a TLS handshake begins immediately. Clients MUST implement the certificate validation mechanism described in [RFC7817]. Once a TLS session is established, message submission protocol data [RFC6409] is exchanged as TLS application data for the remainder of the TCP connection. (Note: the "submissions" service name is defined in section 10.3 of this document, and follows the usual convention that the name of a service layered on top of Implicit TLS consists of the name of the service as used without TLS, with an "s" appended.)

The STARTTLS mechanism on port 587 is relatively widely deployed due to the situation with port 465 (discussed in Section 7.3). This differs from IMAP and POP services where Implicit TLS is more widely

deployed on servers than STARTTLS. It is desirable to migrate core protocols used by MUA software to Implicit TLS over time for consistency as well as the additional reasons discussed in Appendix A. However, to maximize use of encryption for submission it is desirable to support both mechanisms for Message Submission over TLS for a transition period of several years. As a result, clients and servers SHOULD implement both STARTTLS on port 587 and Implicit TLS on port 465 for this transition period. Note that there is no significant difference between the security properties of STARTTLS on port 587 and Implicit TLS on port 465 if the implementations are correct and both client and server are configured to require successful negotiation of TLS prior to message submission.

Note that the "submissions" port provides access to a Mail Submission Agent (MSA) as defined in [RFC6409] so requirements and recommendations for MSAs in that document apply to the submissions port, including the requirement to implement SMTP AUTH [RFC4954].

See Section 5.5 and Section 4.2 for additional information on client certificate authentication. See Section 7.3 for port registration information.

3.4. Implicit TLS Connection Closure for POP, IMAP and SMTP Submission

When a client or server wishes to close the connection, it SHOULD initiate the exchange of TLS close alerts before TCP connection termination. The client MAY, after sending a TLS close alert, gracefully close the TCP connection (e.g. call the close() function on the TCP socket or otherwise issue a TCP CLOSE ([RFC0793] section 3.5) without waiting for a TLS response from the server.

4. Use of TLS by Mail Access Services and Message Submission Services

The following requirements and recommendations apply to Mail Access Services and Mail Submission Services:

- Mail Service Providers (MSPs) that support POP, IMAP, and/or Message Submission, MUST support TLS access for those services.

- Other services than POP, IMAP and/or Message Submission provided by MSPs SHOULD support TLS access, and MUST support TLS access for those services which support authentication via username and password.

- MSPs that support POP, IMAP, and/or Message Submission, SHOULD provide and support instances of those services which use Implicit TLS. (See Section 3.)

- For compatibility with existing MUAs and existing MUA configurations, MSPs SHOULD also, in the near term, provide instances of these services which support STARTTLS. This will permit legacy MUAs to discover new availability of TLS capability on servers, and may increase use of TLS by such MUAs. However, servers SHOULD NOT advertise STARTTLS if use of the STARTTLS command by a client is likely to fail (for example, if the server has no server certificate configured.)

- MSPs SHOULD advertise their Mail Access Services and Mail Submission Services using DNS SRV records according to [RFC6186]. (In addition to making correct configuration easier for MUAs, this provides a way by which MUAs can discover when an MSP begins to offer TLS-based services.) Services supporting TLS SHOULD be advertised in preference to cleartext services (if offered). In addition, services using Implicit TLS SHOULD be advertised in preference to services supporting STARTTLS (if offered). (See also Section 4.5.)

- MSPs SHOULD deprecate use of cleartext Mail Access Services and Mail Submission Services as soon as practicable. (See Section 4.1.)

- MSPs currently supporting such use of cleartext SMTP (on port 25) as a means of message submission by their users (whether or not requiring authentication) SHOULD transition their users to using TLS (either Implicit TLS or STARTTLS) as soon as practicable.

- Mail services MUST support TLS 1.2 or later.

- All Mail services SHOULD implement the recommended TLS cipher suites described in [RFC7525] or a future BCP or standards track revision of that document.

- Mail services currently supporting SSL 2.x, SSL 3.0, or TLS 1.0 SHOULD transition their users to later versions of TLS, and discontinue support for those versions of SSL and TLS, as soon as practicable.

- Mail Submission Servers accepting mail using TLS SHOULD include the TLS ciphersuite of the session in which the mail was received, in the Received field of the outgoing message. (See Section 4.3.)

- All Mail services implementing TLS SHOULD log TLS cipher information along with any connection or authentication logs that they maintain.

Additional considerations and details appear below.

4.1. Deprecation of Services Using Cleartext and TLS Versions < 1.1

The specific means employed for deprecation of cleartext Mail Access Services and Mail Submission Services MAY vary from one MSP to the next in light of their user communities' needs and constraints. For example, an MSP MAY implement a gradual transition in which, over time, more and more users are forbidden to authenticate to cleartext instances of these services, thus encouraging those users to migrate to Implicit TLS. Access to cleartext services should eventually be either disabled, or limited strictly for use by legacy systems which cannot be upgraded.

After a user's ability to authenticate to a service using cleartext is revoked, the server denying such access MUST NOT provide any indication over a cleartext channel of whether the user's authentication credentials were valid. An attempt to authenticate as such a user using either invalid credentials or valid credentials MUST both result in the same indication of access being denied.

Also, users previously authenticating with passwords sent as cleartext SHOULD be required to change those passwords when migrating to TLS, if the old passwords were likely to have been compromised. (For any large community of users using public Internet to access mail without encryption, compromise of at least some of those passwords should be assumed.)

Transition of users from SSL or TLS 1.0 to later versions of TLS MAY be accomplished by a means similar to that described above. There are multiple ways to accomplish this. One way is for the server to refuse a ClientHello message from any client sending a ClientHello.version field corresponding to any version of SSL or TLS 1.0. Another way is for the server to accept ClientHello messages from some client versions that it does not wish to support, but later refuse to allow the user to authenticate. The latter method may provide a better indication to the user of the reason for the failure but (depending on the protocol and method of authentication used) may also risk exposure of the user's password over an channel which is known to not provide adequate confidentiality.

It is RECOMMENDED that new users be required to use TLS version 1.1 or greater from the start. However an MSP may find it necessary to make exceptions to accommodate some legacy systems which support only earlier versions of TLS, or only cleartext.

4.2. Mail Server Use of Client Certificate Authentication

Mail servers MAY implement client certificate authentication on the Implicit TLS port. Servers MUST NOT request a client certificate during the TLS handshake unless the server is configured to accept some client certificates as sufficient for authentication and the server has the ability to determine a mail server authorization identity matching such certificates. How to make this determination is presently implementation specific.

If the server accepts the client's certificate as sufficient for authorization, it MUST enable the SASL EXTERNAL [RFC4422] mechanism. An IMAPS server MAY issue a PREAUTH greeting instead of enabling SASL EXTERNAL.

4.3. Recording TLS Cipher Suite in Received Header

The ESMTPS transmission type [RFC3848] provides trace information that can indicate TLS was used when transferring mail. However, TLS usage by itself is not a guarantee of confidentiality or security. The TLS cipher suite provides additional information about the level of security made available for a connection. This defines a new SMTP "tls" Received header additional-registered-clause that is used to record the TLS cipher suite that was negotiated for the connection. This clause SHOULD be included whenever a Submission server generates a Received header field for a message received via TLS. The value included in this additional clause SHOULD be the registered cipher suite name (e.g., TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256) included in the TLS cipher suite registry. In the event the implementation does not know the name of the cipher suite (a situation that should be remedied promptly), a four-digit hexadecimal cipher suite identifier MAY be used. In addition, the Diffie-Hellman group name associated with the ciphersuite MAY be included (when applicable and known) following the ciphersuite name. The ABNF for the field follows:

```
tls-cipher-clause  =  CFWS "tls" FWS tls-cipher [ CFWS "group" FWS dh-group ]

tls-cipher         =  tls-cipher-name / tls-cipher-hex

tls-cipher-name    =  ALPHA *(ALPHA / DIGIT / "_")
; as registered in IANA cipher suite registry

tls-cipher-hex     =  "0x" 4HEXDIG

dh-group           = ALPHA *(ALPHA / DIGIT / "_" / "-")
; as registered in IANA TLS Supported Groups Registry
```

4.4. TLS Server Certificate Requirements

MSPs MUST maintain valid server certificates for all servers. See [RFC7817] for the recommendations and requirements necessary to achieve this.

If a protocol server provides service for more than one mail domain, it MAY use a separate IP address for each domain and/or a server certificate that advertises multiple domains. This will generally be necessary unless and until it is acceptable to impose the constraint that the server and all clients support the Server Name Indication extension to TLS [RFC6066]. Mail servers supporting SNI need to support the post-SRV hostname to interoperate with MUAs that have not implemented RFC 6186. For more discussion of this problem, see section 5.1 of [RFC7817].

4.5. Recommended DNS records for mail protocol servers

This section discusses not only the DNS records that are recommended, but also implications of DNS records for server configuration and TLS server certificates.

4.5.1. MX records

It is recommended that MSPs advertise MX records for handling of inbound mail (instead of relying entirely on A or AAAA records), and that those MX records be signed using DNSSEC [RFC4033]. This is mentioned here only for completeness, as handling of inbound mail is out of scope for this document.

4.5.2. SRV records

MSPs SHOULD advertise SRV records to aid MUAs in determination of proper configuration of servers, per the instructions in [RFC6186].

MSPs SHOULD advertise servers that support Implicit TLS in preference to those which support cleartext and/or STARTTLS operation.

4.5.3. DNSSEC

All DNS records advertised by an MSP as a means of aiding clients in communicating with the MSP’s servers, SHOULD be signed using DNSSEC if and when the parent DNS zone supports doing so.

4.5.4. TLSA records

MSPs SHOULD advertise TLSA records to provide an additional trust anchor for public keys used in TLS server certificates. However, TLSA records MUST NOT be advertised unless they are signed using DNSSEC.

4.6. Changes to Internet Facing Servers

When an MSP changes the Internet Facing Servers providing mail access and mail submission services, including SMTP-based spam/virus filters, it is generally necessary to support the same and/or a newer version of TLS and the same security directives that were previously advertised.

5. Use of TLS by Mail User Agents

The following requirements and recommendations apply to Mail User Agents:

- MUAs SHOULD be capable of using DNS SRV records to discover Mail Access Services and Mail Submission Services that are advertised by a MSP for an account being configured. Other means of discovering server configuration information (e.g. a database maintained by the MUA vendor) MAY also be supported. (See Section 5.1 for more information.)

- MUAs SHOULD be configurable to require a minimum level of confidentiality for any particular Mail Account, and refuse to exchange information via any service associated with that Mail Account if the session does not provide that minimum level of confidentiality. (See Section 5.2.)

- MUAs MUST NOT treat a session as meeting a minimum level of confidentiality if the server’s TLS certificate cannot be validated. (See Section 5.3.)

- MUAs MAY impose other minimum confidentiality requirements in the future, e.g. in order to discourage use of TLS versions or cryptographic algorithms in which weaknesses have been discovered.

- MUAs SHOULD provide a prominent indication of the level of confidentiality associated with an account configuration that is appropriate for the user interface (for example, a "lock" icon or changed background color for a visual interface, or some sort of audible indication for an audio user interface), at appropriate times and/or locations in order to inform the user of the confidentiality of the communications associated with that

account. For example, this might be done whenever (a) prompting the user for authentication credentials, (b) the user is composing mail that will be sent to a particular submission server, (c) a list of accounts is displayed (particularly if the user can select from that list to read mail), or (d) the user is requesting to view or update any configuration data that will be stored on a remote server. If, however, an MUA provides such an indication, it MUST NOT indicate confidentiality for any connection that does not at least use TLS 1.1 with certificate verification and also meet the minimum confidentiality requirements associated with that account.

- MUAs MUST implement TLS 1.2 [RFC5246] or later. Earlier TLS and SSL versions MAY also be supported so long as the MUA requires at least TLS 1.1 [RFC4346] when accessing accounts that are configured to impose minimum confidentiality requirements.

- All MUAs SHOULD implement the recommended TLS cipher suites described in [RFC7525] or a future BCP or standards track revision of that document.

- MUAs that are configured to not require minimum confidentiality for one or more accounts SHOULD detect when TLS becomes available on those accounts (using [RFC6186] or other means), and offer to upgrade the account to require TLS.

Additional considerations and details appear below.

5.1. Use of SRV records in Establishing Configuration

This section updates [RFC6186] by changing the preference rules and adding a new SRV service label _submissions._tcp to refer to Message Submission with Implicit TLS.

User-configurable MUAs SHOULD support use of [RFC6186] for account setup. However, when using configuration information obtained by this method, MUAs SHOULD ignore advertised services that do not satisfy minimum confidentiality requirements, unless the user has explicitly requested reduced confidentiality. This will have the effect of causing the MUA to default to ignoring advertised configurations that do not support TLS, even when those advertised configurations have a higher priority than other advertised configurations.

When using [RFC6186] configuration information, Mail User Agents SHOULD NOT automatically establish new configurations that do not require TLS for all servers, unless there are no advertised configurations using TLS. If such a configuration is chosen, prior

to attempting to authenticate to the server or use the server for message submission, the MUA SHOULD warn the user that traffic to that server will not be encrypted and that it will therefore likely be intercepted by unauthorized parties. The specific wording is to be determined by the implementation, but it should adequately capture the sense of risk given the widespread incidence of mass surveillance of email traffic.

Similarly, a MUA MUST NOT attempt to "test" a particular mail account configuration by submitting the user's authentication credentials to a server, unless a TLS session meeting minimum confidentiality levels has been established with that server. If minimum confidentiality requirements have not been satisfied, the MUA must explicitly warn the user that his password may be exposed to attackers before testing the new configuration.

When establishing a new configuration for connecting to an IMAP, POP, or SMTP submission server, based on SRV records, an MUA SHOULD either verify that the SRV records are signed using DNSSEC, or that the target FQDN of the SRV record matches the original server FQDN for which the SRV queries were made. If the target FQDN is not in the queried domain, the MUA SHOULD verify with the user that the SRV target FQDN is suitable for use, before executing any connections to the host. (See [RFC6186] section 6).

An MUA MUST NOT consult SRV records to determine which servers to use on every connection attempt, unless those SRV records are signed by DNSSEC and have a valid signature. However, an MUA MAY consult SRV records from time to time to determine if an MSP's server configuration has changed, and alert the user if it appears that this has happened. This can also serve as a means to encourage users to upgrade their configurations to require TLS if and when their MSPs support it.

5.2. Minimum Confidentiality Level

MUAs SHOULD, by default, require a minimum level of confidentiality for services accessed by each account. For MUAs supporting the ability to access multiple mail accounts, this requirement SHOULD be configurable on a per-account basis.

The default minimum expected level of confidentiality for all new accounts MUST require successful validation of the server's certificate and SHOULD require negotiation of TLS version 1.1 or greater. (Future revisions to this specification may raise these requirements or impose additional requirements to address newly-discovered weaknesses in protocols or cryptographic algorithms.)

MUAs MAY permit the user to disable this minimum confidentiality requirement during initial account configuration, or subsequently editing an account configuration, but MUST warn users that such a configuration will not assure privacy for either passwords or messages.

An MUA which is configured to require a minimum level of confidentiality for a mail account MUST NOT attempt to perform any operation other than capability discovery, or STARTTLS for servers not using Implicit TLS, unless the minimum level of confidentiality is provided by that connection.

MUAs SHOULD NOT allow users to easily access or send mail via an connection, or authenticate to any service using a password, if that account is configured to impose minimum confidentiality requirements and that connection does not meet all of those requirements. An example of "easily access" would be to display a dialog informing the user that the security requirements of the account were not met by the connection, but allowing the user to "click through" to send mail or access the service anyway. Experience indicates that users presented with such an option often "click through" without understanding the risks that they're accepting by doing so. Furthermore, users who frequently find the need to "click through" to use an insecure connection may become conditioned to do so as a matter of habit, before considering whether the risks are reasonable in each specific instance.

An MUA which is not configured to require a minimum level of confidentiality for a mail account SHOULD still attempt to connect to the services associated with that account using the most secure means available, e.g. by using Implicit TLS or STARTTLS.

5.3. Certificiate Validation

MUAs MUST validate TLS server certificates according to [RFC7817] and PKIX [RFC5280].

MUAs MAY also support DANE [RFC6698] as a means of validating server certificates in order to meet minimum confidentiality requirements.

MUAs MAY support use of certificate pinning but MUST NOT consider a connection in which the server's authenticity relies on certificate pinning, as providing the minimum level of confidentiality. (See Section 5.4.)

5.4. Certificate Pinning

During account setup, the MUA will identify servers that provide account services such as mail access and mail submission (the previous section describes one way to do this). The certificates for these servers are verified using the rules described in [RFC7817] and PKIX [RFC5280]. In the event the certificate does not validate due to an expired certificate, lack of appropriate chain of trust, or lack of identifier match, the MUA MAY offer to create a persistent binding between that certificate and the saved host name for the server, for use when accessing that account's servers. This is called certificate pinning.

(Note: This use of the term "certificate pinning" means something subtly different than "HTTP Public Key Pinning" [RFC7469]. The dual use of the same term is confusing, but unfortunately both uses are well-established.)

Certificate pinning is only appropriate during mail account setup and MUST NOT be offered as an option in response to a failed certificate validation for an existing mail account. An MUA that allows certificate pinning MUST NOT allow a certificate pinned for one account to validate connections for other accounts. An MUA that allows certificate pinning MUST also allow a user to undo the pinning, i.e. to revoke trust in a certificate that has previously been pinned.

A pinned certificate is subject to a man-in-the-middle attack at account setup time, and typically lacks a mechanism to automatically revoke or securely refresh the certificate. Note also that a man-in-the-middle attack at account setup time will expose the user's password to the attacker (if a password is used). Therefore use of a pinned certificate does not meet the requirement for a minimum confidentiality level, and an MUA MUST NOT indicate to the user that the such confidentiality is provided. Additional advice on certificate pinning is present in [RFC6125].

5.5. Client Certificate Authentication

MUAs MAY implement client certificate authentication on the Implicit TLS port. An MUA MUST NOT provide a client certificate during the TLS handshake unless the server requests one and the MUA has been authorized to use that client certificate with that account. Having the end-user explicitly configure a client certificate for use with a given account is sufficient to meet this requirement. However, installing a client certificate for use with one account MUST NOT automatically authorize use of that certificate with other accounts. This is not intended to prohibit site-specific authorization

mechanisms, such as a site-administrator-controlled mechanism to authorize use of a client certificate with a given account, or a domain-name matching mechanism.

Note: The requirement that the server request a certificate is just a restatement of the TLS protocol rules, e.g. [RFC5246] section 7.4.6. The requirement that the client not send a certificate not known to be acceptable to the server is pragmatic in multiple ways: the current TLS protocol provides no way for the client to know which of potentially multiple certificates it should use; also, when the client sends a certificate it is potentially disclosing its identity (or its user's identity) to both the server and to any party with access to the transmission medium, perhaps unnecessarily and for no useful purpose.

A client supporting client certificate authentication with Implicit TLS MUST implement the SASL EXTERNAL [RFC4422] mechanism using the appropriate authentication command (AUTH for POP3 [RFC5034], AUTH for SMTP Submission [RFC4954], AUTHENTICATE for IMAP [RFC3501]).

6. Considerations related to Anti-Virus/Anti-Spam Software and Services

There are multiple ways to connect an Anti-Virus and/or Anti-Spam (AVAS) service to a mail server. Some mechanisms, such as the de-facto milter protocol, are out of scope for this specification. However, some services use an SMTP relay proxy that intercepts mail at the application layer to perform a scan and proxy or forward to another MTA. Deploying AVAS services in this way can cause many problems [RFC2979] including direct interference with this specification, and other forms of confidentiality or security reduction. An AVAS product or service is considered compatible with this specification if all IMAP, POP and SMTP-related software (including proxies) it includes are compliant with this specification.

Note that end-to-end email encryption prevents AVAS software and services from using email content as part of a spam or virus assessment. Furthermore, while a minimum confidentiality level can prevent a man-in-the-middle from introducing spam or virus content between the MUA and Submission server, it does not prevent other forms of client or account compromise. Use of AVAS services for submitted email therefore remains necessary.

7. IANA Considerations

7.1. POP3S Port Registration Update

IANA is asked to update the registration of the TCP well-known port 995 using the following template ([RFC6335]):

```
Service Name: pop3s
Transport Protocol: TCP
Assignee: IETF <iesg@ietf.org>
Contact: IESG <iesg@ietf.org>
Description: POP3 over TLS protocol
Reference: RFC XXXX (this document once published)
Port Number: 995
```

7.2. IMAPS Port Registration Update

IANA is asked to update the registration of the TCP well-known port 993 using the following templates ([RFC6335]):

```
Service Name: imaps
Transport Protocol: TCP
Assignee: IETF <iesg@ietf.org>
Contact: IESG <iesg@ietf.org>
Description: IMAP over TLS protocol
Reference: RFC XXXX (this document once published)
Port Number: 993
```

No changes to existing UDP port assignments for pop3s or imaps are being requested.

7.3. Submissions Port Registration

IANA is asked to assign an alternate usage of TCP port 465 in addition to the current assignment using the following template ([RFC6335]):

```
Service Name: submissions
Transport Protocol: TCP
Assignee: IETF <iesg@ietf.org>
Contact: IESG <iesg@ietf.org>
Description: Message Submission over TLS protocol
Reference: RFC XXXX (this document once published)
Port Number: 465
```

This is a one-time procedural exception to the rules in RFC 6335. This requires explicit IESG approval and does not set a precedent. Note: Since the purpose of this alternate usage assignment is to align with widespread existing practice, and there is no known usage

of UDP port 465 for message submission over TLS, IANA is not being asked to assign an alternate usage of UDP port 465.

Historically, port 465 was briefly registered as the "smtps" port. This registration made no sense as the SMTP transport MX infrastructure has no way to specify a port, so port 25 is always used. As a result, the registration was revoked and was subsequently reassigned to a different service. In hindsight, the "smtps" registration should have been renamed or reserved rather than revoked. Unfortunately, some widely deployed mail software interpreted "smtps" as "submissions" [RFC6409] and used that port for email submission by default when an end-user requests security during account setup. If a new port is assigned for the submissions service, email software will either continue with unregistered use of port 465 (leaving the port registry inaccurate relative to de-facto practice and wasting a well-known port), or confusion between the de-facto and registered ports will cause harmful interoperability problems that will deter use of TLS for message submission. The authors believe both of these outcomes are less desirable than a wart in the registry documenting real-world usage of a port for two purposes. Although STARTTLS-on-port-587 has deployed, it has not replaced deployed use of Implicit TLS submission on port 465.

7.4. Additional registered clauses for Received fields

Per the provisions in [RFC5321], IANA is requested to add two additional-registered-clauses for Received fields as defined in Section 4.3 of this document:

- "tls" indicating the TLS cipher used (if applicable), and
- "group" indicating the Diffie-Hellman group used with the TLS cipher (if applicable)

The descriptions and syntax of these additional clauses are in Section 4.3 of this document.

8. Security Considerations

This entire document is about security considerations. In general, this is targeted to improve mail confidentiality and to mitigate threats external to the email system such as network-level snooping or interception; this is not intended to mitigate active attackers who have compromised service provider systems.

Implementers should be aware that use of client certificates with TLS 1.2 reveals the user’s identity to any party with ability to read packets from the transmission medium, and therefore may compromise

the user's privacy. There seems to be no easy fix with TLS 1.2 or earlier versions other than to avoid presenting client certificates except when there is explicit authorization to do so. TLS 1.3 [I-D.ietf-tls-tls13] appears to reduce the privacy risk somewhat.

9. References

9.1. Normative References

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC1939] Myers, J. and M. Rose, "Post Office Protocol - Version 3", STD 53, RFC 1939, DOI 10.17487/RFC1939, May 1996, <https://www.rfc-editor.org/info/rfc1939>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, DOI 10.17487/RFC3207, February 2002, <https://www.rfc-editor.org/info/rfc3207>.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, DOI 10.17487/RFC3501, March 2003, <https://www.rfc-editor.org/info/rfc3501>.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC5034] Siemborski, R. and A. Menon-Sen, "The Post Office Protocol (POP3) Simple Authentication and Security Layer (SASL) Authentication Mechanism", RFC 5034, DOI 10.17487/RFC5034, July 2007, <https://www.rfc-editor.org/info/rfc5034>.

[RFC5068] Hutzler, C., Crocker, D., Resnick, P., Allman, E., and T. Finch, "Email Submission Operations: Access and Accountability Requirements", BCP 134, RFC 5068, DOI 10.17487/RFC5068, November 2007, <https://www.rfc-editor.org/info/rfc5068>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC6186] Daboo, C., "Use of SRV Records for Locating Email Submission/Access Services", RFC 6186, DOI 10.17487/RFC6186, March 2011, <https://www.rfc-editor.org/info/rfc6186>.

[RFC6409] Gellens, R. and J. Klensin, "Message Submission for Mail", STD 72, RFC 6409, DOI 10.17487/RFC6409, November 2011, <https://www.rfc-editor.org/info/rfc6409>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <https://www.rfc-editor.org/info/rfc6698>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[RFC7672] Dukhovni, V. and W. Hardaker, "SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)", RFC 7672, DOI 10.17487/RFC7672, October 2015, <https://www.rfc-editor.org/info/rfc7672>.

[RFC7817] Melnikov, A., "Updated Transport Layer Security (TLS) Server Identity Check Procedure for Email-Related Protocols", RFC 7817, DOI 10.17487/RFC7817, March 2016, <https://www.rfc-editor.org/info/rfc7817>.

9.2. Informative References

[I-D.ietf-tls-tls13]
Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-21 (work in progress), July 2017.

[I-D.ietf-uta-mta-sts]
Margolis, D., Risher, M., Ramakrishnan, B., Brotman, A., and J. Jones, "SMTP MTA Strict Transport Security (MTA-STS)", draft-ietf-uta-mta-sts-09 (work in progress), September 2017.

[RFC2595] Newman, C., "Using TLS with IMAP, POP3 and ACAP", RFC 2595, DOI 10.17487/RFC2595, June 1999, <https://www.rfc-editor.org/info/rfc2595>.

[RFC2979] Freed, N., "Behavior of and Requirements for Internet Firewalls", RFC 2979, DOI 10.17487/RFC2979, October 2000, <https://www.rfc-editor.org/info/rfc2979>.

[RFC3848] Newman, C., "ESMTP and LMTP Transmission Types Registration", RFC 3848, DOI 10.17487/RFC3848, July 2004, <https://www.rfc-editor.org/info/rfc3848>.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, DOI 10.17487/RFC4346, April 2006, <https://www.rfc-editor.org/info/rfc4346>.

[RFC4422] Melnikov, A., Ed. and K. Zeilenga, Ed., "Simple Authentication and Security Layer (SASL)", RFC 4422, DOI 10.17487/RFC4422, June 2006, <https://www.rfc-editor.org/info/rfc4422>.

[RFC4954] Siemborski, R., Ed. and A. Melnikov, Ed., "SMTP Service Extension for Authentication", RFC 4954, DOI 10.17487/RFC4954, July 2007, <https://www.rfc-editor.org/info/rfc4954>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <https://www.rfc-editor.org/info/rfc6335>.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <https://www.rfc-editor.org/info/rfc7469>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

Appendix A. Design Considerations

This section is not normative.

The first version of this was written independently from draft-moore-email-tls-00.txt; subsequent versions merge ideas from both drafts.

One author of this document was also the author of RFC 2595 that became the standard for TLS usage with POP and IMAP, and the other author was perhaps the first to propose that idea. In hindsight both authors now believe that that approach was a mistake. At this point the authors believe that while anything that makes it easier to deploy TLS is good, the desirable end state is that these protocols always use TLS, leaving no need for a separate port for cleartext operation except to support legacy clients while they continue to be used. The separate port model for TLS is inherently simpler to implement, debug and deploy. It also enables a "generic TLS load-balancer" that accepts secure client connections for arbitrary foo-over-TLS protocols and forwards them to a server that may or may not support TLS. Such load-balancers cause many problems because they violate the end-to-end principle and the server loses the ability to

log security-relevant information about the client unless the protocol is designed to forward that information (as this specification does for the cipher suite). However, they can result in TLS deployment where it would not otherwise happen which is a sufficiently important goal that it overrides the problems.

Although STARTTLS appears only slightly more complex than separate-port TLS, we again learned the lesson that complexity is the enemy of security in the form of the STARTTLS command injection vulnerability (CERT vulnerability ID #555316). Although there's nothing inherently wrong with STARTTLS, the fact it resulted in a common implementation error (made independently by multiple implementers) suggests it is a less secure architecture than Implicit TLS.

Section 7 of RFC 2595 critiques the separate-port approach to TLS. The first bullet was a correct critique. There are proposals in the http community to address that, and use of SRV records as described in RFC 6186 resolves that critique for email. The second bullet is correct as well, but not very important because useful deployment of security layers other than TLS in email is small enough to be effectively irrelevant. (Also it's less correct than it used to be because "export" ciphersuites are no longer supported in modern versions of TLS.) The third bullet is incorrect because it misses the desirable option of "use and latch-on TLS if available". The fourth bullet may be correct, but is not a problem yet with current port consumption rates. The fundamental error was prioritizing a perceived better design based on a mostly valid critique over real-world deployability. But getting security and confidentiality facilities actually deployed is so important it should trump design purity considerations.

Port 465 is presently used for two purposes: for submissions by a large number of clients and service providers and for the "urd" protocol by one vendor. Actually documenting this current state is controversial as discussed in the IANA considerations section. However, there is no good alternative. Registering a new port for submissions when port 465 is widely used for that purpose already will just create interoperability problems. Registering a port that's only used if advertised by an SRV record (RFC 6186) would not create interoperability problems but would require all client and server deployments and software to change significantly which is contrary to the goal of promoting more TLS use. Encouraging use of STARTTLS on port 587 would not create interoperability problems, but is unlikely to have impact on current undocumented use of port 465 and makes the guidance in this document less consistent. The remaining option is to document the current state of the world and support future use of port 465 for submission as this increases consistency and ease-of-deployment for TLS email submission.

Appendix B. Change Log

Changes since draft-ietf-uta-email-deep-07:

- After discussion with the WG in Prague, removed BCP language and once again made unambiguous that this is intended as a standards-track document.

- Server implementations now MUST implement TLS 1.2, consistent with RFC 7525. MUAs may still consider a TLS 1.1 session as meeting minimum confidentiality requirements.

- MSPs now MUST support TLS for POP, IMAP, Submission, and any other services that use username/password authentication.

- Added text to clarify the purpose of recommending that MSPs use DNS SRV records to advertise services.

- Changed text about MUAs not blindly trusting unsigned SRV records, to instead restate RFC 6186 requirements.

Changes since draft-ietf-uta-email-deep-06:

- On the recommendation of one of the co-chairs and some working group members, rewrote document with the intended status of BCP. This involved removing a great deal of text that consisted essentially of new protocol specification, especially the STS features, on the theory that a BCP should base its recommendations on current practice, and that new protocol features should be subject to the interoperability test requirements associated with normal standards-track documents.

Changes since draft-ietf-uta-email-deep-05:

- Clarify throughout that the confidentiality assurance level associated with a mail account is a minimum level; attempt to distinguish this from the current confidentiality level provided by a connection between client and server.

- Change naming for confidentiality assurance levels: instead of "high" or "no" confidence, assign numbers 1 and 0 to them respectively. This because it seems likely that in the not-too-distant future, what was defined in -05 as "high" confidence will be considered insufficient, and calling that "high" confidence will become misleading. For example, relying entirely on a list of trusted CAs to validate server certificates from arbitrary parties, appears to be less and less reliable in practice at thwarting MITM attacks.

- Clarify that if some services associated with a mail account don't meet the minimum confidentiality assurance level assigned to that account, other services that do meet that minimum confidentiality assurance level may continue to be used.

- Clarify that successful negotiation of at least TLS version 1.1 is required as a condition of meeting confidentiality assurance level 1.

- Clarify that validation of a server certificate using either DANE or PKIX is sufficient to meet the certificate validation requirement of confidentiality assurance level 1.

- Clarify that minimum confidentiality assurance levels are separate from security directives, and that the requirements of both mechanisms must be met.

- Explicitly cite an example that a security directive of tls-version=1.2 won't be saved if the currently negotiated tls-version is 1.1. (This example already appeared a bit later in the text, but for author KM it seemed to make the mechanism clearer to use this example earlier.)

- Clarify some protocol examples as to whether PKIX or DANE was used to verify a server's certificate.

- Remove most references to DEEP as the conversion from DEEP to MUA-STS seemed incomplete, but kept the DEEP command for use in POP3 on the assumption that author CN wanted it that way.

- Removed most references to "latch" and derivative words.

- Added pkix+dane as a value for the tls-cert directive, to indicate (from a server) that both PKIX and DANE validation will be supported, or (from a client) that both PKIX and DANE were used to validate a certificate. Also clarified what each of any, pkix, dane, and pkix+dane mean when advertised by a server and in particular that tls-cert=any provides no assurance of future PKIX verifiability in contrast to tls-cert=pkix or tls-cert=pkix+dane. It seemed important to support the ability to evolve to using multiple trust anchors for certificate validation, but also to allow servers to have the option to migrate from PKIX to DANE if that made sense for them. This change seemed less disruptive than either defining additional directives, or allowing multiple instances of the same directive with different values to appear in the same advertisement.

- Clarify interaction of this specification with anti-virus / anti-spam mechanisms.

Changes since draft-ietf-uta-email-deep-04:

- Swap sections 5.1 and 5.3 ("Email Security Tags" and "Server DEEP Status") as that order may aid understanding of the model. Also rewrote parts of these two sections to try to make the model clearer.
- Add text about versioning of security tags to make the model clearer.
- Add example of security tag upgrade.
- Convert remaining mention of TLS 1.0 to TLS 1.1.
- Change document title from DEEP to MUA STS to align with SMTP relay STS.
  - Slight updates to abstract and introductions.
  - Rename security latches/tags to security directives.
  - Rename server DEEP status to STS policy.
  - Change syntax to use directive-style HSTS syntax.
- Make HSTS reference normative.
- Remove SMTP DSN header as that belongs in SMTP relay STS document.

Changes since draft-ietf-uta-email-deep-03:

- Add more references to ietf-uta-email-tls-certs in implementation requirements section.
- Replace primary reference to RFC 6125 with ietf-uta-email-tls-certs, so move RFC 6125 to informative list for this specification.

Changes since draft-ietf-uta-email-deep-02:

- Make reference to design considerations explicit rather than "elsewhere in this document".
- Change provider requirement so SMTP submission services are separate from SMTP MTA services as opposed to the previous

phrasing that required the servers be separate (which is too restrictive).

- Update DANE SMTP reference

Changes since draft-ietf-uta-email-deep-01:

- Change text in tls11 and tls12 registrations to clarify certificate rules, including additional PKIX and DANE references.
- Change from tls10 to tls11 (including reference) as the minimum.
- Fix typo in example 5.
- Remove open issues section; enough time has passed so not worth waiting for more input.

Changes since draft-ietf-uta-email-deep-00:

- Update and clarify abstract
- use term confidentiality instead of privacy in most cases.
- update open issues to request input for missing text.
- move certificate pinning sub-section to account setup section and attempt to define it more precisely.
- Add note about end-to-end encryption in AVAS section.
- swap order of DNSSEC and TLSA sub-sections.
- change meaning of 'tls10' and 'tls12' latches to require certificate validation.
- Replace cipher suite advice with reference to RFC 7525. Change examples to use TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as cipher suite.
- Add text to update IMAP, POP3 and Message Submission standards with newer TLS advice.
- Add clearer text in introduction that this does not cover SMTP relay.
- Update references to uta-tls-certs.

- Add paragraph to Implicit TLS for SMTP Submission section recommending that STARTTLS also be implemented.

Changes since draft-newman-email-deep-02:

- Changed "privacy assurance" to "confidentiality assurance"
- Changed "low privacy assurance" to "no confidentiality assurance"
- Attempt to improve definition of confidentiality assurance level.
- Add SHOULD indicate when MUA is showing list of mail accounts.
- Add SHOULD NOT latch tls10, tls12 tags until TLS negotiated.
- Removed sentence about deleting and re-creating the account in latch failure section.
- Remove use of word "fallback" with respect to TLS version negotiation.
- Added bullet about changes to Internet facing servers to MSP section.
- minor wording improvements based on feedback

Changes since -01:

- Updated abstract, introduction and document structure to focus more on mail user agent privacy assurance.
- Added email account privacy section, also moving section on account setup using SRV records to that section.
- Finished writing IANA considerations section
- Remove provisional concept and instead have server explicitly list security tags clients should latch.
- Added note that rules for the submissions port follow the same rules as those for the submit port.
- Reference and update advice in [RFC5068].
- Fixed typo in Client Certificate Authentication section.
- Removed tls-pfs security latch and all mention of perfect forward secrecy as it was controversial.

- Added reference to HSTS.

Changes since -00:

- Rewrote introduction to merge ideas from draft-moore-email-tls-00.
- Added Implicit TLS section, Account configuration section and IANA port registration updates based on draft-moore-email-tls-00.
- Add protocol details necessary to standardize implicit TLS for POP/IMAP/submission, using ideas from draft-melnikov-pop3-over-tls.
- Reduce initial set of security tags based on feedback.
- Add deep status concept to allow a window for software updates to be backed out before latches make that problematic, as well as to provide service providers with a mechanism they can use to assist customers in the event of a privacy failure.
- Add DNS SRV section from draft-moore-email-tls-00.
- Write most of the missing IANA considerations section.
- Rewrite most of implementation requirements section based more on draft-moore-email-tls-00. Remove new cipher requirements for now because those may be dealt with elsewhere.

Appendix C. Acknowledgements

Thanks to Ned Freed for discussion of the initial latch concepts in this document. Thanks to Alexey Melnikov for draft-melnikov-pop3-over-tls-02, which was the basis of the POP3 Implicit TLS text. Thanks to Russ Housley, Alexey Melnikov and Dan Newman for review feedback. Thanks to Paul Hoffman for interesting feedback in initial conversations about this idea.

Authors' Addresses

Keith Moore
Windrock, Inc.
PO Box 1934
Knoxville, TN 37901
US

Email: moore@network-heretics.com

Chris Newman
Oracle
440 E. Huntington Dr., Suite 400
Arcadia, CA 91006
US

Email: chris.newman@oracle.com

Using TLS in Applications
Internet-Draft
Intended status: Standards Track
Expires: December 18, 2018

D. Margolis
M. Risher
Google, Inc
B. Ramakrishnan
Yahoo!, Inc
A. Brotman
Comcast, Inc
J. Jones
Microsoft, Inc
June 16, 2018

SMTP MTA Strict Transport Security (MTA-STS)
draft-ietf-uta-mta-sts-21

Abstract

SMTP Mail Transfer Agent Strict Transport Security (MTA-STS) is a mechanism enabling mail service providers to declare their ability to receive Transport Layer Security (TLS) secure SMTP connections, and to specify whether sending SMTP servers should refuse to deliver to MX hosts that do not offer TLS with a trusted server certificate.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 18, 2018.

Copyright Notice

Table of Contents

1. Introduction 3
1.1. Terminology 3
2. Related Technologies 4
3. Policy Discovery 4
3.1. MTA-STS TXT Records 4
3.2. MTA-STS Policies 6
3.3. HTTPS Policy Fetching 9
3.4. Policy Selection for Smart Hosts and Subdomains 10
4. Policy Validation 10
4.1. MX Host Validation 11
4.2. Recipient MTA Certificate Validation 11
5. Policy Application 11
5.1. Policy Application Control Flow 12
6. Reporting Failures 12
7. Interoperability Considerations 13
7.1. SNI Support 13
7.2. Minimum TLS Version Support 13
8. Operational Considerations 14
8.1. Policy Updates 14
8.2. Policy Delegation 14
8.3. Removing MTA-STS 15
8.4. Preserving MX Candidate Traversal 16
9. IANA Considerations 16
9.1. Well-Known URIs Registry 16
9.2. MTA-STS TXT Record Fields 16
9.3. MTA-STS Policy Fields 17
10. Security Considerations 17
10.1. Obtaining a Signed Certificate 17
10.2. Preventing Policy Discovery 18
10.3. Denial of Service 18
10.4. Weak Policy Constraints 19
10.5. Compromise of the Web PKI System 19
11. Contributors 20
12. References 20
12.1. Normative References 20
12.2. Informative References 22
Appendix A. MTA-STS example record & policy 23
Appendix B. Message delivery pseudocode 23
Authors' Addresses 26

1. Introduction

The STARTTLS extension to SMTP [RFC3207] allows SMTP clients and hosts to negotiate the use of a TLS channel for encrypted mail transmission.

While this opportunistic encryption protocol by itself provides a high barrier against passive man-in-the-middle traffic interception, any attacker who can delete parts of the SMTP session (such as the "250 STARTTLS" response) or who can redirect the entire SMTP session (perhaps by overwriting the resolved MX record of the delivery domain) can perform downgrade or interception attacks.

This document defines a mechanism for recipient domains to publish policies, via a combination of DNS and HTTPS, specifying:

- whether MTAs sending mail to this domain can expect PKIX-authenticated TLS support
- what a conforming client should do with messages when TLS cannot be successfully negotiated

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14] [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

We also define the following terms for further use in this document:

- MTA-STS Policy: A commitment by the Policy Domain to support PKIX [RFC5280] authenticated TLS for the specified MX hosts.
- Policy Domain: The domain for which an MTA-STS Policy is defined. This is the next-hop domain; when sending mail to "alice@example.com" this would ordinarily be "example.com", but this may be overridden by explicit routing rules (as described in Section 3.4, "Policy Selection for Smart Hosts and Subdomains").
- Policy Host: The HTTPS host which serves the MTA-STS Policy for a Policy Domain. Rules for constructing the hostname are described in Section 3.2, "MTA-STS Policies".
- Sender: The SMTP Mail Transfer Agent sending an email message.

- ABNF: Augmented Backus-Naur Form, a syntax for formally specifying syntax, defined in [RFC5234] and [RFC7405].

2. Related Technologies

The DANE TLSA record [RFC7672] is similar, in that DANE is also designed to upgrade unauthenticated encryption or plaintext transmission into authenticated, downgrade-resistant encrypted transmission. DANE requires DNSSEC [RFC4033] for authentication; the mechanism described here instead relies on certificate authorities (CAs) and does not require DNSSEC, at a cost of risking malicious downgrades. For a thorough discussion of this trade-off, see Section 10, "Security Considerations".

In addition, MTA-STS provides an optional testing-only mode, enabling soft deployments to detect policy failures; partial deployments can be achieved in DANE by deploying TLSA records only for some of a domain's MXs, but such a mechanism is not possible for the per-domain policies used by MTA-STS.

The primary motivation of MTA-STS is to provide a mechanism for domains to ensure transport security even when deploying DNSSEC is undesirable or impractical. However, MTA-STS is designed not to interfere with DANE deployments when the two overlap; in particular, senders who implement MTA-STS validation MUST NOT allow a "valid" or "testing"-only MTA-STS validation to override a failing DANE validation.

3. Policy Discovery

MTA-STS policies are distributed via HTTPS from a "well-known" [RFC5785] path served within the Policy Domain, and their presence and current version are indicated by a TXT record at the Policy Domain. These TXT records additionally contain a policy "id" field, allowing sending MTAs to check the currency of a cached policy without performing an HTTPS request.

To discover if a recipient domain implements MTA-STS, a sender need only resolve a single TXT record. To see if an updated policy is available for a domain for which the sender has a previously cached policy, the sender need only check the TXT record's version "id" against the cached value.

3.1. MTA-STS TXT Records

The MTA-STS TXT record is a TXT record with the name "_mta-sts" at the Policy Domain. For the domain "example.com", this record would

be "_mta-sts.example.com". MTA-STS TXT records MUST be US-ASCII, semicolon-separated key/value pairs containing the following fields:

- "v": (plain-text, required). Currently only "STSv1" is supported.
- "id": (plain-text, required). A short string used to track policy updates. This string MUST uniquely identify a given instance of a policy, such that senders can determine when the policy has been updated by comparing to the "id" of a previously seen policy. There is no implied ordering of "id" fields between revisions.

An example TXT record is as below:

```
_mta-sts.example.com.  IN TXT "v=STSv1; id=20160831085700Z;"
```

The formal definition of the "_mta-sts" TXT record, defined using ABNF ([RFC7405]), is as follows:

```
sts-text-record = sts-version 1*(sts-field-delim sts-field)
                  [sts-field-delim]

sts-field       = sts-id /                ; Note that sts-id record
                  sts-extension           ; is required.

sts-field-delim = *WSP ";" *WSP

sts-version     = %s"v=STSv1"

sts-id          = %s"id=" 1*32(ALPHA / DIGIT)     ; id=...

sts-extension   = sts-ext-name "=" sts-ext-value  ; name=value

sts-ext-name    = (ALPHA / DIGIT)
                  *31(ALPHA / DIGIT / "_" / "-" / ".")

sts-ext-value   = 1*(%x21-3A / %x3C / %x3E-7E)
                  ; chars excluding "=", ";", SP, and CTLs
```

The TXT record MUST begin with sts-version field, and the order of other fields is not significant. If multiple TXT records for "_mta-sts" are returned by the resolver, records which do not begin with "v=STSv1;" are discarded. If the number of resulting records is not one, or if the resulting record is syntactically invalid, senders MUST assume the recipient domain does not have an available MTA-STS policy and skip the remaining steps of policy discovery. (Note that absence of a usable TXT record is not by itself sufficient to remove a sender’s previously cached policy for the Policy Domain, as discussed in Section 5.1, "Policy Application Control Flow".) If the

resulting TXT record contains multiple strings, then the record MUST be treated as if those strings are concatenated together without adding spaces.

3.2. MTA-STS Policies

The policy itself is a set of key/value pairs (similar to [RFC5322] header fields) served via the HTTPS GET method from the fixed [RFC5785] "well-known" path of ".well-known/mta-sts.txt" served by the Policy Host. The Policy Host DNS name is constructed by prepending "mta-sts" to the Policy Domain.

Thus for a Policy Domain of "example.com" the full URL is "https://mta-sts.example.com/.well-known/mta-sts.txt".

When fetching a policy, senders SHOULD validate that the media type is "text/plain" to guard against cases where webservers allow untrusted users to host non-text content (typically, HTML or images) at a user-defined path. All parameters other than charset=utf-8 or charset=us-ascii are ignored. Additional "Content-Type" parameters are also ignored.

This resource contains the following CRLF-separated key/value pairs:

- "version": Currently only "STSv1" is supported.

- "mode": One of "enforce", "testing", or "none", indicating the expected behavior of a sending MTA in the case of a policy validation failure. See Section 5, "Policy Application." for more details about the three modes.

- "max_age": Max lifetime of the policy (plain-text non-negative integer seconds, maximum value of 31557600). Well-behaved clients SHOULD cache a policy for up to this value from last policy fetch time. To mitigate the risks of attacks at policy refresh time, it is expected that this value typically be in the range of weeks or greater.

- "mx": Allowed MX patterns. One or more patterns matching allowed MX hosts for the Policy Domain. As an example,

```
mx: mail.example.com <CRLF>
mx: *.example.net
```

indicates that mail for this domain might be handled by MX "mail.example.com" or any MX at "example.net". Valid patterns can be either fully specified names ("example.com") or suffixes prefixed by a wildcard ("*.example.net"). If a policy specifies more than one

MX, each MX MUST have its own "mx:" key, and each MX key/value pair MUST be on its own line in the policy file. In the case of Internationalized Domain Names ([RFC5891]), the "mx" value MUST specify the Punycode-encoded A-label [RFC3492] to match against, and not the Unicode-encoded U-label. The full semantics of certificate validation (including the use of wildcard patterns) are described in Section 4.1, "MX Host Validation."

An example policy is as below:

```
version: STSv1
mode: enforce
mx: mail.example.com
mx: *.example.net
mx: backupmx.example.com
max_age: 604800
```

The formal definition of the policy resource, defined using [RFC7405], is as follows:

```
sts-policy-record        = sts-policy-field *WSP
                           *(sts-policy-term sts-policy-field *WSP)
                           [sts-policy-term]

sts-policy-field         = sts-policy-version /      ; required once
                           sts-policy-mode    /      ; required once
                           sts-policy-max-age /      ; required once

                           sts-policy-term /
                           ; required at least once, except when
                           ; mode is "none"

                           sts-policy-extension      ; other fields

sts-policy-field-delim   = ":" *WSP

sts-policy-version       = sts-policy-version-field sts-policy-field-delim
                           sts-policy-version-value

sts-policy-version-field = %s"version"

sts-policy-version-value = %s"STSv1"

sts-policy-mode          = sts-policy-mode-field sts-policy-field-delim
                           sts-policy-mode-value

sts-policy-mode-field    = %s"mode"
```

```
sts-policy-mode-value    =  %s"testing" / %s"enforce" / %s"none"

sts-policy-mx            = sts-policy-mx-field sts-policy-field-delim
                           sts-policy-mx-value

sts-policy-mx-field      = %s"mx"

sts-policy-mx-value      = ["."] Domain

sts-policy-mx-label      = sts-policy-alphanum /
                           sts-policy-alphanum *(sts-policy-alphanum / "-")
                           sts-policy-alphanum

sts-policy-mx-toplabel   = ALPHA / ALPHA *(sts-policy-alphanum / "-")
                           sts-policy-alphanum

sts-policy-max-age       = sts-policy-max-age-field sts-policy-field-delim
                           sts-policy-max-age-value

sts-policy-max-age-field = %s"max_age"

sts-policy-max-age-value = 1*10(DIGIT)

sts-policy-extension     = sts-policy-ext-name    ; additional
                           sts-policy-field-delim ; extension
                           sts-policy-ext-value   ; fields

sts-policy-ext-name      = (sts-policy-alphanum)
                           *31(sta-policy-alphanum / "_" / "-" / ".")

sts-policy-term          = LF / CRLF

sts-policy-ext-value     = sts-policy-vchar
                           [*(%x20 / sts-policy-vchar)
                           sts-policy-vchar]
                           ; chars, including UTF-8 [@!RFC3629],
                           ; excluding CTLs and no
                           ; leading/trailing spaces

sts-policy-alphanum     = ALPHA / DIGIT

sts-policy-vchar        = %x21-7E / UTF8-2 / UTF8-3 / UTF8-4

UTF8-2          =   <Defined in Section 4 of [@!RFC3629]>

UTF8-3          =   <Defined in Section 4 of [@!RFC3629]>

UTF8-4          =   <Defined in Section 4 of [@!RFC3629]>
```

Domain = <see RFC 5321 4.1.2>

Parsers MUST accept TXT records and policy files which are syntactically valid (i.e., valid key/value pairs separated by semi-colons for TXT records), possibly containing additional key/value pairs not specified in this document, in which case unknown fields SHALL be ignored. If any non-repeated field--i.e., all fields excepting "mx"--is duplicated, all entries except for the first SHALL be ignored.

3.3. HTTPS Policy Fetching

Policy bodies are, as described above, retrieved by sending MTAs via HTTPS [RFC2818]. During the TLS handshake initiated to fetch a new or updated policy from the Policy Host, the Policy Host HTTPS server MUST present a X.509 certificate which is valid for the "mta-sts" DNS-ID ([RFC6125]) (e.g., "mta-sts.example.com") as described below, chain to a root CA that is trusted by the sending MTA, and be non-expired. It is expected that sending MTAs use a set of trusted CAs similar to those in widely deployed Web browsers and operating systems. See [RFC5280] for more details about certificate verification.

The certificate is valid for the Policy Host (i.e., "mta-sts" prepended to the Policy Domain) with respect to the rules described in [RFC6125], with the following application-specific considerations:

- Matching is performed only against the DNS-ID identifiers.
- DNS domain names in server certificates MAY contain the wildcard character '*' as the complete left-most label within the identifier.

The certificate MAY be checked for revocation via the Online Certificate Status Protocol (OCSP) [RFC6960], certificate revocation lists (CRLs), or some other mechanism.

Policies fetched via HTTPS are only valid if the HTTP response code is 200 (OK). HTTP 3xx redirects MUST NOT be followed, and HTTP caching (as specified in [RFC7234]) MUST NOT be used.

Senders may wish to rate-limit the frequency of attempts to fetch the HTTPS endpoint even if a valid TXT record for the recipient domain exists. In the case that the HTTPS GET fails, implementers SHOULD limit further attempts to a period of five minutes or longer per version ID, to avoid overwhelming resource-constrained recipients with cascading failures.

Senders MAY impose a timeout on the HTTPS GET and/or a limit on the maximum size of the response body to avoid long delays or resource exhaustion during attempted policy updates. A suggested timeout is one minute, and a suggested maximum policy size 64 kilobytes; policy hosts SHOULD respond to requests with a complete policy body within that timeout and size limit.

If a valid TXT record is found but no policy can be fetched via HTTPS (for any reason), and there is no valid (non-expired) previously-cached policy, senders MUST continue with delivery as though the domain has not implemented MTA-STS.

Conversely, if no "live" policy can be discovered via DNS or fetched via HTTPS, but a valid (non-expired) policy exists in the sender's cache, the sender MUST apply that cached policy.

Finally, to mitigate the risk of persistent interference with policy refresh, as discussed in-depth in Section 10, MTAs SHOULD proactively refresh cached policies before they expire; a suggested refresh frequency is once per day. To enable administrators to discover problems with policy refresh, MTAs SHOULD alert administrators (through the use of logs or similar) when such attempts fail, unless the cached policy mode is "none".

3.4. Policy Selection for Smart Hosts and Subdomains

When sending mail via a "smart host"--an administratively configured intermediate SMTP relay, which is different from the message recipient's server as determined from DNS --compliant senders MUST treat the smart host domain as the policy domain for the purposes of policy discovery and application. This specification does not provide a means of associating policies with addresses that employ Address Literals [RFC5321].

When sending mail to a mailbox at a subdomain, compliant senders MUST NOT attempt to fetch a policy from the parent zone. Thus for mail sent to "user@mail.example.com", the policy can be fetched only from "mail.example.com", not "example.com".

4. Policy Validation

When sending to an MX at a domain for which the sender has a valid and non-expired MTA-STS policy, a sending MTA honoring MTA-STS MUST check whether:

1. At least one of the policy's "mx" patterns matches the selected MX host, as described in Section 4.1, "MX Host Validation".

2. The recipient mail server supports STARTTLS and offers a PKIX-based TLS certificate, during TLS handshake, which is valid for that host, as described in Section 4.2, "Recipient MTA Certificate Validation".

When these conditions are not met, a policy is said to fail to validate. This section does not dictate the behavior of sending MTAs when the above conditions are not met; see Section 5, "Policy Application" for a description of sending MTA behavior when policy validation fails.

4.1. MX Host Validation

A receiving candidate MX host is valid according to an applied MTA-STS policy if the MX record name matches one or more of the "mx" fields in the applied policy. Matching is identical to the rules given in [RFC6125], with restriction that the wildcard character "*" may only be used to match the entire left-most label in the presented identifier. Thus the mx pattern "*.example.com" matches "mail.example.com" but not "example.com" or "foo.bar.example.com".

4.2. Recipient MTA Certificate Validation

The certificate presented by the receiving MTA MUST not be expired, and MUST chain to a root CA that is trusted by the sending MTA. The certificate MUST have a subject alternative name (SAN, [RFC5280]) with a DNS-ID ([RFC6125]) matching the host name, per the rules given in [RFC6125]. The MX’s certificate MAY also be checked for revocation via OCSP [RFC6960], CRLs [RFC6818], or some other mechanism.

5. Policy Application

When sending to an MX at a domain for which the sender has a valid, non-expired MTA-STS policy, a sending MTA honoring MTA-STS applies the result of a policy validation failure one of two ways, depending on the value of the policy "mode" field:

1. "enforce": In this mode, sending MTAs MUST NOT deliver the message to hosts which fail MX matching or certificate validation, or do not support STARTTLS.

2. "testing": In this mode, sending MTAs which also implement the TLSRPT specification [I-D.ietf-uta-smtp-tlsrpt] merely send a report indicating policy application failures (so long as TLSRPT is also implemented by the recipient domain).

3. "none": In this mode, sending MTAs should treat the policy domain as though it does not have any active policy; see Section 8.3, "Removing MTA-STS", for use of this mode value.

When a message fails to deliver due to an "enforce" policy, a compliant MTA MUST NOT permanently fail to deliver messages before checking, via DNS, for the presence of an updated policy at the Policy Domain. (In all cases, MTAs SHOULD treat such failures as transient errors and retry delivery later.) This allows implementing domains to update long-lived policies on the fly.

5.1. Policy Application Control Flow

An example control flow for a compliant sender consists of the following steps:

1. Check for a cached policy whose time-since-fetch has not exceeded its "max_age". If none exists, attempt to fetch a new policy (perhaps asynchronously, so as not to block message delivery). Optionally, sending MTAs may unconditionally check for a new policy at this step.

2. For each candidate MX, in order of MX priority, attempt to deliver the message. If a policy is present with an "enforce" mode, when attempting to deliver to each candidate MX, ensure STARTTLS support and host identity validity as described in Section 4, "Policy Validation". If a candidate fails validation, continue to the next candidate (if there is one).

3. A message delivery MUST NOT be permanently failed until the sender has first checked for the presence of a new policy (as indicated by the "id" field in the "_mta-sts" TXT record). If a new policy is not found, existing rules for the case of temporary message delivery failures apply (as discussed in [RFC5321] section 4.5.4.1).

6. Reporting Failures

MTA-STS is intended to be used along with TLSRPT [I-D.ietf-uta-smtp-tlsrpt] in order to ensure implementing domains can detect cases of both benign and malicious failures, and to ensure that failures that indicate an active attack are discoverable. As such, senders who also implement TLSRPT SHOULD treat the following events as reportable failures:

- HTTPS policy fetch failures when a valid TXT record is present.

- Policy fetch failures of any kind when a valid policy exists in the policy cache, except if that policy's mode is "none".

- Delivery attempts in which a contacted MX does not support STARTTLS or does not present a certificate which validates according to the applied policy, except if that policy's mode is "none".

7. Interoperability Considerations

7.1. SNI Support

To ensure that the server sends the right certificate chain, the SMTP client MUST have support for the TLS SNI extension [RFC6066]. When connecting to a HTTP server to retrieve the MTA-STS policy, the SNI extension MUST contain the name of the policy host (e.g., "mta-sts.example.com"). When connecting to an SMTP server, the SNI extension MUST contain the MX hostname.

HTTP servers used to deliver MTA-STS policies MAY rely on SNI to determine which certificate chain to present to the client. HTTP servers MUST respond with a certificate chain that matches the policy hostname or abort the TLS handshake if unable to do so. Clients that do not send SNI information may not see the expected certificate chain.

SMTP servers MAY rely on SNI to determine which certificate chain to present to the client. However servers that have one identity and a single matching certificate do not require SNI support. Servers MUST NOT enforce the use of SNI by clients, as the client may be using unauthenticated opportunistic TLS and may not expect any particular certificate from the server. If the client sends no SNI extension or sends an SNI extension for an unsupported server name, the server MUST simply send a fallback certificate chain of its choice. The reason for not enforcing strict matching of the requested SNI hostname is that MTA-STS TLS clients may be typically willing to accept multiple server names but can only send one name in the SNI extension. The server's fallback certificate may match a different name that is acceptable to the client, e.g., the original next-hop domain.

7.2. Minimum TLS Version Support

MTAs supporting MTA-STS MUST have support for TLS version 1.2 [RFC5246] or higher. The general TLS usage guidance in [RFC7525] SHOULD be followed.

8. Operational Considerations

8.1. Policy Updates

Updating the policy requires that the owner make changes in two places: the "_mta-sts" TXT record in the Policy Domain's DNS zone and at the corresponding HTTPS endpoint. As a result, recipients should expect a policy will continue to be used by senders until both the HTTPS and TXT endpoints are updated and the TXT record's TTL has passed.

In other words, a sender who is unable to successfully deliver a message while applying a cache of the recipient's now-outdated policy may be unable to discover that a new policy exists until the DNS TTL has passed. Recipients SHOULD therefore ensure that old policies continue to work for message delivery during this period of time, or risk message delays.

Recipients SHOULD also update the HTTPS policy body before updating the TXT record; this ordering avoids the risk that senders, seeing a new TXT record, mistakenly cache the old policy from HTTPS.

8.2. Policy Delegation

Domain owners commonly delegate SMTP hosting to a different organization, such as an ISP or a Web host. In such a case, they may wish to also delegate the MTA-STS policy to the same organization which can be accomplished with two changes.

First, the Policy Domain must point the "_mta-sts" record, via CNAME, to the "_mta-sts" record maintained by the hosting organization. This allows the hosting organization to control update signaling.

Second, the Policy Domain must point the "well-known" policy location to the hosting organization. This can be done either by setting the "mta-sts" record to an IP address or CNAME specified by the hosting organization and by giving the hosting organization a TLS certificate which is valid for that host, or by setting up a "reverse proxy" (also known as a "gateway") server that serves as the Policy Domain's policy the policy currently served by the hosting organization.

For example, given a user domain "user.example" hosted by a mail provider "provider.example", the following configuration would allow policy delegation:

DNS:

```
_mta-sts.user.example.  IN CNAME _mta-sts.provider.example.
```

Policy:

```
> GET /.well-known/mta-sts.txt Host: mta-sts.user.example
< HTTP/1.1 200 OK  # Response proxies content from
                   # https://mta-sts.provider.example
```

Note that in all such cases, the policy endpoint ("https://mta-sts.user.example/.well-known/mta-sts.txt" in this example) must still present a certificate valid for the Policy Host ("mta-sts.user.example"), and not for that host at the provider's domain ("mta-sts.provider.example").

Note that while sending MTAs MUST NOT use HTTP caching when fetching policies via HTTPS, such caching may nonetheless be useful to a reverse proxy configured as described in this section. An HTTPS policy endpoint expecting to be proxied for multiple hosted domains--as with a large mail hosting provider or similar--may wish to indicate an HTTP Cache-Control "max-age" response directive (as specified in [RFC7234]) of 60 seconds as a reasonable value to save reverse proxies an unnecessarily high-rate of proxied policy fetching.

8.3. Removing MTA-STS

In order to facilitate clean opt-out of MTA-STS by implementing policy domains, and to distinguish clearly between failures which indicate attacks and those which indicate such opt-outs, MTA-STS implements the "none" mode, which allows validated policies to indicate authoritatively that the policy domain wishes to no longer implement MTA-STS and may, in the future, remove the MTA-STS TXT and policy endpoints entirely.

A suggested workflow to implement such an opt out is as follows:

1. Publish a new policy with "mode" equal to "none" and a small "max_age" (e.g., one day).

2. Publish a new TXT record to trigger fetching of the new policy.

3. When all previously served policies have expired--normally this is the time the previously published policy was last served plus that policy's "max_age", but note that older policies may have been served with a greater "max_age", allowing overlapping policy caches--safely remove the TXT record and HTTPS endpoint.

8.4. Preserving MX Candidate Traversal

Implementors of send-time MTA-STS validation in mail transfer agents should take note of the risks of modifying the logic of traversing MX candidate lists. Because an MTA-STS policy can be used to prefilter invalid MX candidates from the MX candidate list, it is tempting to implement a "two-pass" model, where MX candidates are first filtered for possible validity according to the MTA-STS policy, and then the remaining candidates attempted in order as without an MTA-STS policy. This may lead to incorrect implementations, such a message loops; implementors are instead recommended to traverse the MX candidate list as usual, and treat invalid candidates as though they were unreachable (i.e., as though there were some transient error when trying to deliver to that candidate).

One consequence of validating MX hosts in order of ordinary candidate traversal is that, in the event that a higher-priority MX is MTA-STS valid and a lower-priority MX is not, senders may never encounter the lower-priority MX, leading to a risk that policy misconfigurations that apply only to "backup" MXes may only be discovered in the case of primary MX failure.

9. IANA Considerations

9.1. Well-Known URIs Registry

A new "well-known" URI as described in Section 3 will be registered in the Well-Known URIs registry as described below:

URI Suffix: mta-sts.txt Change Controller: IETF

9.2. MTA-STS TXT Record Fields

IANA is requested to create a new registry titled "MTA-STS TXT Record Fields". The initial entries in the registry are:

Field Name	Description	Reference
v	Record version	Section 3.1 of RFC XXX
id	Policy instance ID	Section 3.1 of RFC XXX

New fields are added to this registry using IANA’s "Expert Review" policy.

9.3. MTA-STS Policy Fields

IANA is requested to create a new registry titled "MTA-STS Policy Fields". The initial entries in the registry are:

Field Name	Description	Reference
version	Policy version	Section 3.2 of RFC XXX
mode	Enforcement behavior	Section 3.2 of RFC XXX
max_age	Policy lifetime	Section 3.2 of RFC XXX
mx	MX identities	Section 3.2 of RFC XXX

New fields are added to this registry using IANA's "Expert Review" policy.

10. Security Considerations

SMTP MTA Strict Transport Security attempts to protect against an active attacker trying to intercept or tamper with mail between hosts that support STARTTLS. There are two classes of attacks considered:

- Foiling TLS negotiation, for example by deleting the "250 STARTTLS" response from a server or altering TLS session negotiation. This would result in the SMTP session occurring over plaintext, despite both parties supporting TLS.

- Impersonating the destination mail server, whereby the sender might deliver the message to an impostor, who could then monitor and/or modify messages despite opportunistic TLS. This impersonation could be accomplished by spoofing the DNS MX record for the recipient domain, or by redirecting client connections intended for the legitimate recipient server (for example, by altering BGP routing tables).

MTA-STS can thwart such attacks only if the sender is able to previously obtain and cache a policy for the recipient domain, and only if the attacker is unable to obtain a valid certificate that complies with that policy. Below, we consider specific attacks on this model.

10.1. Obtaining a Signed Certificate

SMTP MTA-STS relies on certificate validation via PKIX based TLS identity checking [RFC6125]. Attackers who are able to obtain a valid certificate for the targeted recipient mail service (e.g., by

compromising a certificate authority) are thus able to circumvent STS authentication.

10.2. Preventing Policy Discovery

Since MTA-STS uses DNS TXT records for policy discovery, an attacker who is able to block DNS responses can suppress the discovery of an MTA-STS Policy, making the Policy Domain appear not to have an MTA-STS Policy. The sender policy cache is designed to resist this attack by decreasing the frequency of policy discovery and thus reducing the window of vulnerability; it is nonetheless a risk that attackers who can predict or induce policy discovery--for example, by inducing a sending domain to send mail to a never-before-contacted recipient while carrying out a man-in-the-middle attack--may be able to foil policy discovery and effectively downgrade the security of the message delivery.

Since this attack depends upon intercepting initial policy discovery, implementers SHOULD prefer policy "max_age" values to be as long as is practical.

Because this attack is also possible upon refresh of a cached policy, implementors SHOULD NOT wait until a cached policy has expired before checking for an update; if senders attempt to refresh the cache regularly (for example, by fetching currently live policy in a background task that runs daily or weekly, regardless of the state of the "_mta_sts" TXT record, and updating their cache’s "max age" accordingly), an attacker would have to foil policy discovery consistently over the lifetime of a cached policy to prevent a successful refresh.

Additionally, MTAs SHOULD alert administrators to repeated policy refresh failures long before cached policies expire (through warning logs or similar applicable mechanisms), allowing administrators to detect such a persistent attack on policy refresh. (However, they should not implement such alerts if the cached policy has a "none" mode, to allow clean MTA-STS removal, as described in Section 8.3.)

Resistance to downgrade attacks of this nature--due to the ability to authoritatively determine "lack of a record" even for non-participating recipients--is a feature of DANE, due to its use of DNSSEC for policy discovery.

10.3. Denial of Service

We additionally consider the Denial of Service risk posed by an attacker who can modify the DNS records for a recipient domain. Absent MTA-STS, such an attacker can cause a sending MTA to cache

invalid MX records, but only for however long the sending resolver caches those records. With MTA-STS, the attacker can additionally advertise a new, long-"max_age" MTA-STS policy with "mx" constraints that validate the malicious MX record, causing senders to cache the policy and refuse to deliver messages once the victim has resecured the MX records.

This attack is mitigated in part by the ability of a victim domain to (at any time) publish a new policy updating the cached, malicious policy, though this does require the victim domain to both obtain a valid CA-signed certificate and to understand and properly configure MTA-STS.

Similarly, we consider the possibility of domains that deliberately allow untrusted users to serve untrusted content on user-specified subdomains. In some cases (e.g., the service Tumblr.com) this takes the form of providing HTTPS hosting of user-registered subdomains; in other cases (e.g. dynamic DNS providers) this takes the form of allowing untrusted users to register custom DNS records at the provider’s domain.

In these cases, there is a risk that untrusted users would be able to serve custom content at the "mta-sts" host, including serving an illegitimate MTA-STS policy. We believe this attack is rendered more difficult by the need for the attacker to also serve the "_mta-sts" TXT record on the same domain--something not, to our knowledge, widely provided to untrusted users. This attack is additionally mitigated by the aforementioned ability for a victim domain to update an invalid policy at any future date.

10.4. Weak Policy Constraints

Even if an attacker cannot modify a served policy, the potential exists for configurations that allow attackers on the same domain to receive mail for that domain. For example, an easy configuration option when authoring an MTA-STS Policy for "example.com" is to set the "mx" equal to "*.example.com"; recipient domains must consider in this case the risk that any user possessing a valid hostname and CA-signed certificate (for example, "dhcp-123.example.com") will, from the perspective of MTA-STS Policy validation, be a valid MX host for that domain.

10.5. Compromise of the Web PKI System

A host of risks apply to the PKI system used for certificate authentication, both of the "mta-sts" HTTPS host’s certificate and the SMTP servers’ certificates. These risks are broadly applicable

within the Web PKI ecosystem and are not specific to MTA-STS; nonetheless, they deserve some consideration in this context.

Broadly speaking, attackers may compromise the system by obtaining certificates under fraudulent circumstances (i.e., by impersonating the legitimate owner of the victim domain), by compromising a Certificate Authority or Delegate Authority's private keys, by obtaining a legitimate certificate issued to the victim domain, and similar.

One approach commonly employed by Web browsers to help mitigate against some of these attacks is to allow for revocation of compromised or fraudulent certificates via OCSP [RFC6960] or CRLs [RFC6818]. Such mechanisms themselves represent tradeoffs and are not universally implemented; we nonetheless recommend implementors of MTA-STS to implement revocation mechanisms which are most applicable to their implementations.

11. Contributors

Wei Chuang Google, Inc weihaw@google.com

Viktor Dukhovni ietf-dane@dukhovni.de

Markus Laber 1&1 Mail & Media Development & Technology GmbH markus.laber@1und1.de

Nicolas Lidzborski Google, Inc nlidz@google.com

Brandon Long Google, Inc blong@google.com

Franck Martin LinkedIn, Inc fmartin@linkedin.com

Klaus Umbach 1&1 Mail & Media Development & Technology GmbH klaus.umbach@1und1.de

12. References

12.1. Normative References

[I-D.ietf-uta-smtp-tlsrpt]
Margolis, D., Brotman, A., Ramakrishnan, B., Jones, J., and M. Risher, "SMTP TLS Reporting", draft-ietf-uta-smtp-tlsrpt-22 (work in progress), May 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, DOI 10.17487/RFC3207, February 2002, <https://www.rfc-editor.org/info/rfc3207>.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, DOI 10.17487/RFC3492, March 2003, <https://www.rfc-editor.org/info/rfc3492>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, DOI 10.17487/RFC5785, April 2010, <https://www.rfc-editor.org/info/rfc5785>.

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/info/rfc7405>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

12.2. Informative References

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <https://www.rfc-editor.org/info/rfc4033>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <https://www.rfc-editor.org/info/rfc5891>.

[RFC6818] Yee, P., "Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 6818, DOI 10.17487/RFC6818, January 2013, <https://www.rfc-editor.org/info/rfc6818>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <https://www.rfc-editor.org/info/rfc6960>.

[RFC7234] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Caching", RFC 7234, DOI 10.17487/RFC7234, June 2014, <https://www.rfc-editor.org/info/rfc7234>.

[RFC7672] Dukhovni, V. and W. Hardaker, "SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)", RFC 7672, DOI 10.17487/RFC7672, October 2015, <https://www.rfc-editor.org/info/rfc7672>.

Appendix A. MTA-STS example record & policy

The owner of "example.com" wishes to begin using MTA-STS with a policy that will solicit reports from senders without affecting how the messages are processed, in order to verify the identity of MXs that handle mail for "example.com", confirm that TLS is correctly used, and ensure that certificates presented by the recipient MX validate.

MTA-STS policy indicator TXT RR:

```
_mta-sts.example.com.  IN TXT "v=STSv1; id=20160831085700Z;"
```

MTA-STS Policy file served as the response body at "https://mta-sts.example.com/.well-known/mta-sts.txt":

```
version: STSv1
mode: testing
mx: mx1.example.com
mx: mx2.example.com
mx: mx.backup-example.com
max_age: 1296000
```

Appendix B. Message delivery pseudocode

Below is pseudocode demonstrating the logic of a compliant sending MTA.

While this pseudocode implementation suggests synchronous policy retrieval in the delivery path, in a working implementation that may be undesirable, and we expect some implementers to instead prefer a background fetch that does not block delivery if no cached policy is present.

```
func isEnforce(policy) {
  // Return true if the policy mode is "enforce".
```

```
}

func isNonExpired(policy) {
  // Return true if the policy is not expired.
}

func tryStartTls(connection) {
  // Attempt to open an SMTP connection with STARTTLS with the MX.
}

func certMatches(connection, host) {
  // Assume a handy function to return check if the server certificate presented
  // in "connection" is valid for "host".
}

func policyMatches(candidate, policy) {
  for mx in policy.mx {
    // Literal match.
    if mx == candidate {
      return true
    }
    // Wildcard matches only the leftmost label.
    // Wildcards must always be followed by a '.'.
    if mx[0] == '*' {
      parts = SplitN(candidate, '.', 2)  // Split on the first '.'.
      if len(parts) > 1 && parts[1] == mx[2:] {
        return true
      }
    }
  }
  return false
}

func tryDeliverMail(connection, message) {
  // Attempt to deliver "message" via "connection".
}

func tryGetNewPolicy(domain) {
  // Check for an MTA-STS TXT record for "domain" in DNS, and return the
  // indicated policy.
}

func cachePolicy(domain, policy) {
  // Store "policy" as the cached policy for "domain".
}

func tryGetCachedPolicy(domain) {
  // Return a cached policy for "domain".
```

```
}

func reportError(error) {
  // Report an error via TLSRPT.
}

func tryMxAccordingTo(message, mx, policy) {
  connection := connect(mx)
  if !connection {
    return false  // Can't connect to the MX so it's not an MTA-STS
                  // error.
  }
  secure := true
  if !policyMatches(mx, policy) {
    secure = false
    reportError(E_HOST_MISMATCH)
  } else if !tryStartTls(connection) {
    secure = false
    reportError(E_NO_VALID_TLS)
  } else if !certMatches(connection, policy) {
    secure = false
    reportError(E_CERT_MISMATCH)
  }
  if secure || !isEnforce(policy) {
    return tryDeliverMail(connection, message)
  }
  return false
}

func tryWithPolicy(message, domain, policy) {
  mxes := getMxForDomain(domain)
  for mx in mxes {
    if tryMxAccordingTo(message, mx, policy) {
      return true
    }
  }
  return false
}

func handleMessage(message) {
  domain := ... // domain part after '@' from recipient
  policy := tryGetNewPolicy(domain)
  if policy {
    cachePolicy(domain, policy)
  } else {
    policy = tryGetCachedPolicy(domain)
  }
  if policy {
```

```
    return tryWithPolicy(message, domain, policy)
  }
  // Try to deliver the message normally (i.e., without MTA-STS).
}
```

Authors' Addresses

Daniel Margolis
Google, Inc

Email: dmargolis@google.com

Mark Risher
Google, Inc

Email: risher@google.com

Binu Ramakrishnan
Yahoo!, Inc

Email: rbinu@yahoo-inc.com

Alexander Brotman
Comcast, Inc

Email: alex_brotman@comcast.com

Janet Jones
Microsoft, Inc

Email: janet.jones@microsoft.com

Using TLS in Applications
Internet-Draft
Intended status: Standards Track
Expires: December 16, 2018

D. Margolis
Google, Inc
A. Brotman
Comcast, Inc
B. Ramakrishnan
Yahoo!, Inc
J. Jones
Microsoft, Inc
M. Risher
Google, Inc
June 14, 2018

SMTP TLS Reporting
draft-ietf-uta-smtp-tlsrpt-23

Abstract

A number of protocols exist for establishing encrypted channels between SMTP Mail Transfer Agents, including STARTTLS, DANE TLSA, and MTA-STS. These protocols can fail due to misconfiguration or active attack, leading to undelivered messages or delivery over unencrypted or unauthenticated channels. This document describes a reporting mechanism and format by which sending systems can share statistics and specific information about potential failures with recipient domains. Recipient domains can then use this information to both detect potential attacks and diagnose unintentional misconfigurations.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 16, 2018.

Table of Contents

1. Introduction 3
1.1. Terminology 4
2. Related Technologies 4
3. Reporting Policy 5
3.1. Example Reporting Policy 7
3.1.1. Report using MAILTO 7
3.1.2. Report using HTTPS 7
4. Reporting Schema 7
4.1. Report Time-frame 8
4.2. Delivery Summary 9
4.2.1. Success Count 9
4.2.2. Failure Count 9
4.3. Result Types 9
4.3.1. Negotiation Failures 10
4.3.2. Policy Failures 10
4.3.3. General Failures 11
4.3.4. Transient Failures 11
4.4. JSON Report Schema 11
4.5. Policy Samples 14
5. Report Delivery 15
5.1. Report Filename 15
5.2. Compression 16
5.3. Email Transport 16
5.3.1. Example Report 17
5.4. HTTPS Transport 18
5.5. Delivery Retry 19
5.6. Metadata Variances 19
6. IANA Considerations 19
6.1. Message headers 19
6.2. Report Type 19
6.3. +gzip Media Type Suffix 20

6.4. application/tlsrpt+json Media Type 21
6.5. application/tlsrpt+gzip Media Type 23
6.6. STARTTLS Validation Result Types 24
7. Security Considerations 24
8. Privacy Considerations 26
9. References 26
9.1. Normative References 26
9.2. Informative References 28
9.3. URIs 29
Appendix A. Example Reporting Policy 30
A.1. Report using MAILTO 30
A.2. Report using HTTPS 30
Appendix B. Example JSON Report 30
Authors' Addresses 32

1. Introduction

The STARTTLS extension to SMTP [RFC3207] allows SMTP clients and hosts to establish secure SMTP sessions over TLS. The protocol design uses an approach that has come to be known as "Opportunistic Security" (OS) [RFC7435]. This method maintains interoperability with clients that do not support STARTTLS, but means that any attacker could potentially eavesdrop on a session. An attacker could perform a downgrade or interception attack by deleting parts of the SMTP session (such as the "250 STARTTLS" response) or redirect the entire SMTP session (perhaps by overwriting the resolved MX record of the delivery domain).

Because such "downgrade attacks" are not necessarily apparent to the receiving MTA, this document defines a mechanism for sending domains to report on failures at multiple stages of the MTA-to-MTA conversation.

Recipient domains may also use the mechanisms defined by MTA-STS [I-D.ietf-uta-mta-sts] or DANE [RFC6698] to publish additional encryption and authentication requirements; this document defines a mechanism for sending domains that are compatible with MTA-STS or DANE to share success and failure statistics with recipient domains.

Specifically, this document defines a reporting schema that covers failures in routing, DNS resolution, STARTTLS negotiation, and both DANE [RFC6698] and MTA-STS [I-D.ietf-uta-mta-sts] policy validation errors, and a standard TXT record that recipient domains can use to indicate where reports in this format should be sent. The report can also serve as a heartbeat that systems are successfully negotiating TLS during sessions as expected.

This document is intended as a companion to the specification for SMTP MTA Strict Transport Security [I-D.ietf-uta-mta-sts], as well as adds reporting abilities for those implementing DANE [RFC7672].

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14] [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

We also define the following terms for further use in this document:

- MTA-STS Policy: A mechanism by which administrators can specify the expected TLS availability, presented identity, and desired actions for a given email recipient domain. MTA-STS is defined in [I-D.ietf-uta-mta-sts].

- DANE Policy: A mechanism by which administrators can use DNSSEC to commit an MTA to support STARTTLS and to publish criteria to be used to validate its presented certificates. DANE for SMTP is defined in [RFC7672], with the base specification in [RFC6698] (updated in [RFC7671].

- TLSRPT Policy: A policy specifying the endpoint to which sending MTAs should deliver reports.

- Policy Domain: The domain against which an MTA-STS or DANE Policy is defined. For MTA-STS this is typically the same as the envelope recipient domain [RFC5321], but when mail is routed to a "smarthost" gateway by local policy, the "smarthost" domain name is used instead. For DANE the Policy Domain is the "TLSA base domain" of the receiving SMTP server as described in RFC7672 [1] and RFC6698 [2].

- Sending MTA: The MTA initiating the relay of an email message.

- Aggregate Report URI (rua): A comma-separated list of locations where the report is to be submitted.

2. Related Technologies

- This document is intended as a companion to the specification for SMTP MTA Strict Transport Security [I-D.ietf-uta-mta-sts].

- SMTP-TLSRPT defines a mechanism for sending domains that are compatible with MTA-STS or DANE to share success and failure

statistics with recipient domains. DANE is defined in [RFC6698] and MTA-STS is defined in [I-D.ietf-uta-mta-sts].

3. Reporting Policy

A domain publishes a record to its DNS indicating that it wishes to receive reports. These SMTP TLSRPT policies are distributed via DNS from the Policy Domain's zone, as TXT records (similar to DMARC policies) under the name "_smtp._tls". For example, for the Policy Domain "example.com", the recipient's TLSRPT policy can be retrieved from "_smtp._tls.example.com".

Policies consist of the following directives:

- "v": This document defines version 1 of TLSRPT, for which this value MUST be equal to "TLSRPTv1". Other versions may be defined in later documents.

- "rua": A URI specifying the endpoint to which aggregate information about policy validation results should be sent (see Section 4, "Reporting Schema", for more information). Two URI schemes are supported: "mailto" and "https". As with DMARC [RFC7489], the policy domain can specify a comma-separated list of URIs.

- In the case of "https", reports should be submitted via POST ([RFC7231]) to the specified URI. Report submitters MAY ignore certificate validation errors when submitting reports via https.

- In the case of "mailto", reports should be submitted to the specified email address ([RFC6068]). When sending failure reports via SMTP, sending MTAs MUST deliver reports despite any TLS-related failures and SHOULD NOT include this SMTP session in the next report. When sending failure reports via HTTPS, sending MTAs MAY deliver reports despite any TLS-related faliures. This may mean that the reports are delivered in the clear. Reports sent via SMTP MUST contain a valid DKIM [RFC6376] signature by the reporting domain. Reports lacking such a signature MUST be ignored by the recipient. DKIM signatures must not use the "l=" attribute to limit the body length used in the signature. The DKIM TXT record must contain the appropriate service type declaration, "s=tlsrpt", and if not present the receiving system SHOULD ignore reports signed using this record.

The formal definition of the "_smtp._tls" TXT record, defined using [RFC5234] & [RFC7405], is as follows:

```
tlsrpt-record     = tlsrpt-version 1*(field-delim tlsrpt-field)
                    [field-delim]

field-delim       = *WSP ";" *WSP

tlsrpt-field      = tlsrpt-rua /        ; Note that the
                    tlsrpt-extension    ; tlsrpt-rua record is
                                        ; required.

tlsrpt-version    = %s"v=TLSRPTv1"

tlsrpt-rua        = %s"rua="
                    tlsrpt-uri *(*WSP "," *WSP tlsrpt-uri)

tlsrpt-uri        = URI
                    ; "URI" is imported from [RFC3986];
                    ; commas (ASCII 0x2C), exclamation
                    ; points (ASCII 0x21), and semicolons
                    ; (ASCII 0x3B) MUST be encoded

tlsrpt-extension  = tlsrpt-ext-name "=" tlsrpt-ext-value

tlsrpt-ext-name   = (ALPHA / DIGIT) *31(ALPHA /
                    DIGIT / "_" / "-" / ".")

tlsrpt-ext-value  = 1*(%x21-3A / %x3C / %x3E-7E)
                    ; chars excluding "=", ";", SP, and control
                    ; chars
```

If multiple TXT records for "_smtp._tls" are returned by the resolver, records which do not begin with "v=TLSRPTv1;" are discarded. If the number of resulting records is not one, senders MUST assume the recipient domain does not implement TLSRPT. If the resulting TXT record contains multiple strings (as described in Section 3.1.3 of [RFC4408]), then the record MUST be treated as if those strings are concatenated together without adding spaces.

The record supports the abillity to declare more than one rua, and if there exists more than one, the reporter MAY attempt to deliver to each of the supported rua destinations. A receiver MAY opt to only attempt delivery to one of the endpoints, however the report SHOULD NOT be considered successfully delivered until one of the endpoints accepts delivery of the report.

Parsers MUST accept TXT records which are syntactically valid (i.e. valid key-value pairs separated by semi-colons) and implementing a superset of this specification, in which case unknown fields SHALL be ignored.

3.1. Example Reporting Policy

3.1.1. Report using MAILTO

```
            _smtp._tls.example.com. IN TXT \
                    "v=TLSRPTv1;rua=mailto:reports@example.com"
```

3.1.2. Report using HTTPS

```
           _smtp._tls.example.com. IN TXT \
                   "v=TLSRPTv1; \
                   rua=https://reporting.example.com/v1/tlsrpt"
```

4. Reporting Schema

The report is composed as a plain text file encoded in the I-JSON format ([RFC7493]).

Aggregate reports contain the following fields:

- Report metadata:
  - The organization responsible for the report
  - Contact information for one or more responsible parties for the contents of the report
  - A unique identifier for the report
  - The reporting date range for the report
- Policy, consisting of:
  - One of the following policy types: (1) The MTA-STS policy applied (as a string) (2) The DANE TLSA record applied (as a string, with each RR entry of the RRset listed and separated by a semicolon) (3) The literal string "no-policy-found", if neither a DANE nor MTA-STS policy could be found.
  - The domain for which the policy is applied
  - The MX host
- Aggregate counts, comprising result type, sending MTA IP, receiving MTA hostname, session count, and an optional additional information field containing a URI for recipients to review further information on a failure type.

Note that the failure types are non-exclusive; an aggregate report may contain overlapping "counts" of failure types when a single send attempt encountered multiple errors. Reporters may report multiple applied policies (for example, an MTA-STS policy and a DANE TLSA record for the same domain and MX). Because of this, even in the case where only a single policy was applied, the "policies" field of the report body MUST be an array and not a singular value.

In the case of multiple failure types, the "failure-details" array would contain multiple entries. Each entry would have its own set of infomation pertaining to that failure type.

4.1. Report Time-frame

The report SHOULD cover a full day, from 0000-2400 UTC. This should allow for easier correlation of failure events. To avoid a Denial of Service against the system processing the reports, the reports should be delivered after some delay, perhaps several hours.

As an example, a sending site might want to introduce a random delay of up to four hours:

```
func generate_sleep_delay() {
  min_delay = 1
  max_delay = 14400
  rand = random(min_delay,max_delay)
  return rand
}

func generate_report(policy_domain) {
  do_rpt_work(policy_domain)
  send_rpt(policy_domain)
}

func generate_tlsrpt() {
  sleep(generate_sleep_delay())
  for policy_domain in list_of_tlsrpt_enabled_domains {
    generate_report(policy_domain)
  }
}
```

A sending site might wish to introduce a random delay per destination site, up to four hours:

```
func generate_sleep_delay() {
  min_delay = 1
  max_delay = 14400
  rand = random(min_delay,max_delay)
  return rand
}

func generate_report(policy_domain) {
  sleep(generate_sleep_delay())
  do_rpt_work(policy_domain)
  send_rpt(policy_domain)
}

func generate_tlsrpt() {
  for policy_domain in list_of_tlsrpt_enabled_domains {
    generate_report(policy_domain)
  }
}
```

4.2. Delivery Summary

4.2.1. Success Count

- "total-successful-session-count": This indicates that the sending MTA was able to successfully negotiate a policy-compliant TLS connection, and serves to provide a "heartbeat" to receiving domains that reporting is functional and tabulating correctly. This field contains an aggregate count of successful connections for the reporting system.

4.2.2. Failure Count

- "total-failure-session-count": This indicates that the sending MTA was unable to successfully establish a connection with the receiving platform. Section 4.3, "Result Types", will elaborate on the failed negotiation attempts. This field contains an aggregate count of failed connections.

4.3. Result Types

The list of result types will start with the minimal set below, and is expected to grow over time based on real-world experience. The initial set is:

4.3.1. Negotiation Failures

- "starttls-not-supported": This indicates that the recipient MX did not support STARTTLS.

- "certificate-host-mismatch": This indicates that the certificate presented did not adhere to the constraints specified in the MTA-STS or DANE policy, e.g. if the MX hostname does not match any identities listed in the Subject Alternate Name (SAN) [RFC5280].

- "certificate-expired": This indicates that the certificate has expired.

- "certificate-not-trusted": This a label that covers multiple certificate related failures that include, but not limited to errors such as untrusted/unknown CAs, certificate name constraints, certificate chain errors etc. When using this declaration, the reporting MTA SHOULD utilize the "failure-reason-code" to provide more information to the receiving entity.

- "validation-failure": This indicates a general failure for a reason not matching a category above. When using this declaration, the reporting MTA SHOULD utilize the "failure-reason-code" to provide more information to the receiving entity.

4.3.2. Policy Failures

4.3.2.1. DANE-specific Policy Failures

- "tlsa-invalid": This indicates a validation error in the TLSA record associated with a DANE policy. None of the records in the RRset were found to be valid.

- "dnssec-invalid": This would indicate that no valid records were returned from the recursive resolver. The request returned with SERVFAIL for the requested TLSA record.

- "dane-required": This indicates that the sending system is configured to require DANE TLSA records for all the MX hosts of the destination domain, but no DNSSEC-validated TLSA records were present for the MX host that is the subject of the report. Mandatory DANE for SMTP is described in section 6 of [RFC7672]. Such policies may be created by mutual agreement between two organizations that frequently exchange sensitive content via email.

4.3.2.2. MTA-STS-specific Policy Failures

- "sts-policy-invalid": This indicates a validation error for the overall MTA-STS policy.

- "sts-webpki-invalid": This indicates that the MTA-STS policy could not be authenticated using PKIX validation.

4.3.3. General Failures

When a negotiation failure can not be categorized into one of the "Negotiation Failures" stated above, the reporter SHOULD use the "validation-failure" category. As TLS grows and becomes more complex, new mechanisms may not be easily categorized. This allows for a generic feedback category. When this category is used, the reporter SHOULD also use the "failure-reason-code" to give some feedback to the receiving entity. This is intended to be a short text field, and the contents of the field should be an error code or error text, such as "X509_V_ERR_UNHANDLED_CRITICAL_CRL_EXTENSION".

4.3.4. Transient Failures

Transient errors due to too-busy network, TCP timeouts, etc. are not required to be reported.

4.4. JSON Report Schema

The JSON schema is derived from the HPKP JSON schema [RFC7469] (cf. Section 3)

```
{
  "organization-name": organization-name,
  "date-range": {
    "start-datetime": date-time,
    "end-datetime": date-time
  },
  "contact-info": email-address,
  "report-id": report-id,
  "policies": [{
    "policy": {
      "policy-type": policy-type,
      "policy-string": policy-string,
      "policy-domain": domain,
      "mx-host": mx-host-pattern
    },
    "summary": {
      "total-successful-session-count": total-successful-session-count,
      "total-failure-session-count": total-failure-session-count
    },
    "failure-details": [
      {
        "result-type": result-type,
        "sending-mta-ip": ip-address,
        "receiving-mx-hostname": receiving-mx-hostname,
        "receiving-mx-helo": receiving-mx-helo,
        "receiving-ip": receiving-ip,
        "failed-session-count": failed-session-count,
        "additional-information": additional-info-uri,
        "failure-reason-code": failure-reason-code
        }
      ]
    }
  ]
}
```

JSON Report Format

- "organization-name": The name of the organization responsible for the report. It is provided as a string.

- "date-time": The date-time indicates the start- and end-times for the report range. It is provided as a string formatted according to Section 5.6, "Internet Date/Time Format", of [RFC3339]. The report should be for a full UTC day, 0000-2400.

- "email-address": The contact information for a responsible party of the report. It is provided as a string formatted according to Section 3.4.1, "Addr-Spec", of [RFC5321].

- "report-id": A unique identifier for the report. Report authors may use whatever scheme they prefer to generate a unique identifier. It is provided as a string.

- "policy-type": The type of policy that was applied by the sending domain. Presently, the only three valid choices are "tlsa", "sts", and the literal string "no-policy-found". It is provided as a string.

- "policy-string": An encoding of the applied policy as a JSON array of strings, whether TLSA record ([RFC6698] section 2.3) or MTA-STS policy. Examples follow in the next section.

- "domain": The Policy Domain is the domain against which the MTA-STS or DANE policy is defined. In the case of Internationalized Domain Names ([RFC5891]), the domain MUST consist of the Punycode-encoded A-labels ([RFC3492]) and not the U-labels.

- "mx-host-pattern": The pattern of MX hostnames from the applied policy. It is provided as a string, and is interpreted in the same manner as the "Checking of Wildcard Certificates" rules in Section 6.4.3 of [RFC6125]. In the case of Internationalized Domain Names ([RFC5891]), the domain MUST consist of the Punycode-encoded A-labels ([RFC3492]) and not the U-labels.

- "result-type": A value from Section 4.3, "Result Types", above.

- "ip-address": The IP address of the sending MTA that attempted the STARTTLS connection. It is provided as a string representation of an IPv4 (see below) or IPv6 ([RFC5952]) address in dot-decimal or colon-hexadecimal notation.

- "receiving-mx-hostname": The hostname of the receiving MTA MX record with which the sending MTA attempted to negotiate a STARTTLS connection.

- "receiving-mx-helo": (optional) The HELO or EHLO string from the banner announced during the reported session.

- "receiving-ip": The destination IP address that was using when creating the outbound session. It is provided as a string representation of an IPv4 (see below) or IPv6 ([RFC5952]) address in dot-decimal or colon-hexadecimal notation.

- "total-successful-session-count": The aggregate count (integer, encoded as a JSON number) of successfully negotiated TLS-enabled connections to the receiving site.

- "total-failure-session-count": The aggregate count (integer, encoded as a JSON number) of failures to negotiate a TLS-enabled connection to the receiving site.

- "failed-session-count": The number of (attempted) sessions that match the relevant "result-type" for this section (integer, encoded as a JSON number).

- "additional-info-uri": An optional URI [RFC3986] pointing to additional information around the relevant "result-type". For example, this URI might host the complete certificate chain presented during an attempted STARTTLS session.

- "failure-reason-code": A text field to include a TLS-related error code or error message.

For report purposes, an IPv4 Address is defined via the following ABNF:

```
IPv4address = dec-octet "." dec-octet "." dec-octet "." dec-octet
dec-octet     = DIGIT                 ; 0-9
              / %x31-39 DIGIT         ; 10-99
              / "1" 2DIGIT            ; 100-199
              / "2" %x30-34 DIGIT     ; 200-249
              / "25" %x30-35          ; 250-255
```

4.5. Policy Samples

Part of the report body includes the policy that is applied when attemping relay to the destination.

For DANE TLSA policies, this is a JSON array of strings each representing the RDATA of a single TLSA resource record as a space-separated list of its four TLSA fields; the fields are in presentation format (defined in [RFC6698] Section 2.2) with no internal spaces or grouping parentheses:

```
[
"3 0 1 1F850A337E6DB9C609C522D136A475638CC43E1ED424F8EEC8513D747D1D085D",
"3 0 1 12350A337E6DB9C6123522D136A475638CC43E1ED424F8EEC8513D747D1D1234"
]
```

For MTA-STS policies, this is an array of JSON strings that represents the policy that is declared by the receiving site,

including any errors that may be present. Note that where there are multiple "mx" values, they must be listed as separate "mx" elements in the policy array, rather as a single nested "mx" sub-array.

```
[
"version: STSv1",
"mode: testing",
"mx: mx1.example.com",
"mx: mx2.example.com",
"mx: mx.backup-example.com",
"max_age: 604800"
]
```

5. Report Delivery

Reports can be delivered either as an email message via SMTP or via HTTP POST.

5.1. Report Filename

The filename is RECOMMENDED to be constructed using the following ABNF:

```
 filename        = sender "!" policy-domain "!" begin-timestamp
                   "!" end-timestamp [ "!" unique-id ] "." extension

 unique-id       = 1*(ALPHA / DIGIT)

 sender          = domain ; From the [RFC5321] that is used
                   ; as the domain for the ‘contact-info‘
                   ; address in the report body

 policy-domain   = domain

 begin-timestamp = 1*DIGIT
                   ; seconds since 00:00:00 UTC January 1, 1970
                   ; indicating start of the time range contained
                   ; in the report

 end-timestamp   = 1*DIGIT
                   ; seconds since 00:00:00 UTC January 1, 1970
                   ; indicating end of the time range contained
                   ; in the report

 extension       = "json" / "json.gz"
```

The extension MUST be "json" for a plain JSON file, or "json.gz" for a JSON file compressed using GZIP.

"unique-id" allows an optional unique ID generated by the Sending MTA to distinguish among multiple reports generated simultaneously by different sources within the same Policy Domain. For example, this is a possible filename for a compressed report to the Policy Domain "example.net" from the Sending MTA "mail.sndr.example.com":

"mail.sndr.example.com!example.net!1470013207!1470186007!001.json.gz"

5.2. Compression

The report SHOULD be subjected to GZIP [RFC1952] compression for both email and HTTPS transport. Declining to apply compression can cause the report to be too large for a receiver to process (a commonly observed receiver limit is ten megabytes); compressing the file increases the chances of acceptance of the report at some compute cost.

5.3. Email Transport

The report MAY be delivered by email. To make the reports machine-parsable for the receivers, we define a top-level media type "multipart/report" with a new parameter "report-type="tlsrpt"". Inside it, there are two parts: The first part is human readable, typically "text/plain", and the second part is machine readable with a new media type defined called "application/tlsrpt+json". If compressed, the report should use the media type "application/tlsrpt+gzip".

In addition, the following two new top level message header fields are defined:

"TLS-Report-Domain: Receiver-Domain"

"TLS-Report-Submitter: Sender-Domain"

The "TLS-Report-Submitter" value MUST match the value found in the [RFC5321] domain from the "contact-info" from the report body. These message headers MUST be included and should allow for easy searching for all reports submitted by a report domain or a particular submitter, for example in IMAP [RFC3501]:

"s SEARCH HEADER "TLS-Report-Domain" "example.com""

It is presumed that the aggregate reporting address will be equipped to process new message header fields and extract MIME parts with the prescribed media type and filename, and ignore the rest. These additional headers SHOULD be included in the DKIM [RFC6376] signature for the message.

The [RFC5322].Subject field for report submissions SHOULD conform to the following ABNF:

```
    tlsrpt-subject = %s"Report" FWS               ; "Report"
                     %s"Domain:" FWS              ; "Domain:"
                     domain-name FWS              ; per [RFC6376]
                     %s"Submitter:" FWS           ; "Submitter:"
                     domain-name FWS              ; per [RFC6376]
                     %s"Report-ID:" FWS           ; "Report-ID:
                     "<" id-left "@" id-right ">" ; per [RFC5322]
                     [CFWS]                       ; per [RFC5322]
                                                  ; (as with FWS)
```

The first domain-name indicates the DNS domain name about which the report was generated. The second domain-name indicates the DNS domain name representing the Sending MTA generating the report. The purpose of the Report-ID: portion of the field is to enable the Policy Domain to identify and ignore duplicate reports that might be sent by a Sending MTA.

For instance, this is a possible Subject field for a report to the Policy Domain "example.net" from the Sending MTA "mail.sender.example.com". It is line-wrapped as allowed by [RFC5322]:

```
Subject: Report Domain: example.net
    Submitter: mail.sender.example.com
    Report-ID: <735ff.e317+bf22029@mailexample.net>
```

5.3.1. Example Report

```
      From: tlsrpt@mail.sender.example.com
          Date: Fri, May 09 2017 16:54:30 -0800
          To: mts-sts-tlsrpt@example.net
          Subject: Report Domain: example.net
              Submitter: mail.sender.example.com
              Report-ID: <735ff.e317+bf22029@example.net>
          TLS-Report-Domain: example.net
          TLS-Report-Submitter: mail.sender.example.com
          MIME-Version: 1.0
          Content-Type: multipart/report; report-type="tlsrpt";
              boundary="----=_NextPart_000_024E_01CC9B0A.AFE54C00"
          Content-Language: en-us

          This is a multipart message in MIME format.

          ------=_NextPart_000_024E_01CC9B0A.AFE54C00
          Content-Type: text/plain; charset="us-ascii"
          Content-Transfer-Encoding: 7bit

          This is an aggregate TLS report from mail.sender.example.com

          ------=_NextPart_000_024E_01CC9B0A.AFE54C00
          Content-Type: application/tlsrpt+gzip
          Content-Transfer-Encoding: base64
          Content-Disposition: attachment;
              filename="mail.sender.example!example.com!
                        1013662812!1013749130.json.gz"

          <gzipped content of report>

      ------=_NextPart_000_024E_01CC9B0A.AFE54C00--
      ...
```

Note that, when sending failure reports via SMTP, sending MTAs MUST NOT honor MTA-STS or DANE TLSA failures.

5.4. HTTPS Transport

The report MAY be delivered by POST to HTTPS. If compressed, the report SHOULD use the media type "application/tlsrpt+gzip", and "application/tlsrpt+json" otherwise (see section Section 6, "IANA Considerations").

The receiving system MUST return a "successful" response from its HTTPS server, typically a 200 or 201 HTTP code [RFC7321]. Other codes could indicate a delivery failure, and may be retried as per local sender policy. The receiving system is not expected to process

reports at receipt time, and MAY store them for processing at a later time.

5.5. Delivery Retry

In the event of a delivery failure, regardless of the delivery method, a sender SHOULD attempt redelivery for up to 24hrs after the initial attempt. As previously stated the reports are optional, so while it is ideal to attempt redelivery, it is not required. If multiple retries are attempted, ideally they SHOULD be done with exponential backoff.

5.6. Metadata Variances

As stated above, there are a variable number of ways to declare information about the data therein. If any of items declared via subject or filename disagree with the report, the report MUST be considered the authoritative source.

6. IANA Considerations

The following are the IANA considerations discussed in this document.

6.1. Message headers

Below is the Internet Assigned Numbers Authority (IANA) Permanent Message Header Field registration information per [RFC3864].

Header field name:	TLS-Report-Domain
Applicable protocol:	mail
Status:	standard
Author/Change controller:	IETF
Specification document(s):	this one

Header field name:	TLS-Report-Submitter
Applicable protocol:	mail
Status:	standard
Author/Change controller:	IETF
Specification document(s):	this one

6.2. Report Type

This document creates a new registry for "report-type" parameter to the Content-Type header field for the "multipart/report" top-level media type defined in [RFC6522].

The registry name is "Report Type Registry", and the procedure for updating the registry will be "Specification Required".

An entry in this registry should contain:

- the report-type being registered
- one or more registered media-types that can be used with this report-type
- the document containing the registration action
- an optional comment

The initial entries are:

Report-Type: tlsrpt Media Type: application/tlsrpt+gzip, application/tlsrpt+json Registered By: [RFCXXXX] Comment: Media types suitable for use with this report-type are defined in Sections 6.4 and 6.5 of [RFCXXXX]

Report-Type: disposition-notification Media Type: message/disposition-notification Registered By: [RFC8098] Section 10

Report-Type: disposition-notification Media Type: message/global-disposition-notification Registered By: [RFC6533] Section 6

Report-Type: delivery-status Media Type: message/delivery-status Registered By: [RFC3464] Appendix D

Report-Type: delivery-status Media Type: message/global-delivery-status Registered By: [RFC6533] Section 6

6.3. +gzip Media Type Suffix

This document registers a new media type suffix "+gzip". The GZIP format is a public domain, cross-platform, interoperable file storage and transfer format, specified in [RFC1952]; it supports compression and is used as the underlying representation by a variety of file formats. The media type "application/gzip" has been registered for such files. The suffix "+gzip" MAY be used with any media type whose representation follows that established for "application/gzip". The media type structured syntax suffix registration form follows:

Type name: GZIP file storage and transfer format

+suffix: +gzip

References: [RFC1952][RFC6713]

Encoding considerations: GZIP is a binary encoding.

Fragment identifier considerations: The syntax and semantics of fragment identifiers specified for +gzip SHOULD be as specified for "application/gzip". (At publication of this document, there is no fragment identification syntax defined for "application/gzip".) The syntax and semantics for fragment identifiers for a specific "xxx/yyy+gzip" SHOULD be processed as follows:

> For cases defined in +gzip, where the fragment identifier resolves per the +gzip rules, then process as specified in +gzip.
>
> For cases defined in +gzip, where the fragment identifier does not resolve per the +gzip rules, then process as specified in "xxx/yyy+gzip".
>
> For cases not defined in +gzip, then process as specified in "xxx/yyy+gzip".

Interoperability considerations: n/a

Security considerations: GZIP format doesn’t provide confidentiality protection. Integrity protection is provided by and Adler-32 checksum, which is not cryptographically strong. See also security considerations of [RFC6713]. Each individual media type registered with a +gzip suffix can have additional security considerations. Additionally, GZIP objects can contain multiple files and associated paths. File paths must be validated when the files are extracted; a malicious file path could otherwise cause the extractor to overwrite application or system files.

Contact: art@ietf.org

Author/Change controller: Internet Engineering Task Force (mailto:iesg@ietf.org).

6.4. application/tlsrpt+json Media Type

This document registers multiple media types, beginning with Table 1 below.

Type	Subtype	File extn	Specification
application	tlsrpt+json	.json	Section 5.3

Table 1: SMTP TLS Reporting Media Type

Type name: application

Subtype name: tlsrpt+json

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Encoding considerations are identical to those specified for the "application/json" media type. See [RFC7493].

Security considerations: Security considerations relating to SMTP TLS Reporting are discussed in Section 7.

Interoperability considerations: This document specifies format of conforming messages and the interpretation thereof.

Published specification: Section 5.3 of this document.

Applications that use this media type: Mail User Agents (MUA) and Mail Transfer Agents.

Additional information:

Magic number(s): n/a

File extension(s): ".json"

Macintosh file type code(s): n/a

Person & email address to contact for further information: See Authors' Addresses section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See Authors' Addresses section.

Change controller: Internet Engineering Task Force (mailto:iesg@ietf.org).

6.5. application/tlsrpt+gzip Media Type

Type	Subtype	File extn	Specification
application	tlsrpt+gzip	.gz	Section 5.3

Table 2: SMTP TLS Reporting Media Type

Type name: application

Subtype name: tlsrpt+gzip

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Binary

Security considerations: Security considerations relating to SMTP TLS Reporting are discussed in Section 7. Security considerations related to gzip compression are discussed in [RFC6713].

Interoperability considerations: This document specifies format of conforming messages and the interpretation thereof.

Published specification: Section 5.3 of this document.

Applications that use this media type: Mail User Agents (MUA) and Mail Transfer Agents.

Additional information:

Magic number(s): The first two bytes are 0x1f, 0x8b.

File extension(s): ".gz"

Macintosh file type code(s): n/a

Person & email address to contact for further information: See Authors' Addresses section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See Authors' Addresses section.

Change controller: Internet Engineering Task Force (mailto:iesg@ietf.org).

6.6. STARTTLS Validation Result Types

This document creates a new registry, "STARTTLS Validation Result Types". The initial entries in the registry are:

Result Type	Desc
"starttls-not-supported"	4.3
"certificate-host-mismatch"	4.3
"certificate-expired"	4.3
"tlsa-invalid"	4.3
"dnssec-invalid"	4.3
"dane-required"	4.3
"certificate-not-trusted"	4.3
"sts-policy-invalid"	4.3
"sts-webpki-invalid"	4.3
"validation-failure"	4.3

The above entries are described in section Section 4.3, "Result Types." New result types can be added to this registry using "Expert Review" IANA registration policy.

7. Security Considerations

SMTP TLS Reporting provides transparency into misconfigurations or attempts to intercept or tamper with mail between hosts who support STARTTLS. There are several security risks presented by the existence of this reporting channel:

- Flooding of the Aggregate report URI (rua) endpoint: An attacker could flood the endpoint with excessive reporting traffic and prevent the receiving domain from accepting additional reports. This type of Denial-of-Service attack would limit visibility into STARTTLS failures, leaving the receiving domain blind to an ongoing attack.

- Untrusted content: An attacker could inject malicious code into the report, opening a vulnerability in the receiving domain. Implementers are advised to take precautions against evaluating the contents of the report.

- Report snooping: An attacker could create a bogus TLSRPT record to receive statistics about a domain the attacker does not own. Since an attacker able to poison DNS is already able to receive counts of SMTP connections (and, absent DANE or MTA-STS policies, actual SMTP message payloads), this does not present a significant new vulnerability.

- Ignoring HTTPS validation when submitting reports: When reporting benign misconfigurations, it is likely that a misconfigured SMTP server may also mean a misconfigured HTTPS server; as a result, reporters who required HTTPS validity on the reporting endpoint may fail to alert administrators about such misconfigurations. Conversely, in the event of an actual attack, an attacker who wished to create a gap in reporting and could intercept HTTPS reports could, just as easily, simply thwart the resolution of the TLSRPT TXT record or establishment of the TCP session to the HTTPS endpoint. Furthermore, such a man-in-the-middle attacker could discover most or all of the metadata exposed in a report merely through passive observation. As a result, we consider the risks of failure to deliver reports on misconfigurations to outweigh those of attackers intercepting reports.

- Reports as DDoS: TLSRPT allows specifying destinations for the reports that are outside the authority of the Policy Domain, which allows domains to delegate processing of reports to a partner organization. However, an attacker who controls the Policy Domain DNS could also use this mechanism to direct the reports to an unwitting victim, flooding that victim with excessive reports. DMARC [RFC7489] defines a solution for verifying delegation to avoid such attacks; the need for this is greater with DMARC, however, because DMARC allows an attacker to trigger reports to a target from an innocent third party by sending that third party mail (which triggers a report from the third party to the target). In the case of TLSRPT, the attacker would have to induce the third party to send the attacker mail in order to trigger reports from the third party to the victim; this reduces the risk of such an attack and the need for a verification mechanism.

Finally, because TLSRPT is intended to help administrators discover man-in-the-middle attacks against transport-layer encryption, including attacks designed to thwart negotiation of encrypted connections (by downgrading opportunistic encryption or, in the case of MTA-STS, preventing discovery of a new MTA-STS policy), we must also consider the risk that an adversary who can induce such a downgrade attack can also prevent discovery of the TLSRPT TXT record (and thus prevent discovery of the successful downgrade attack). Administrators are thus encouraged to deploy TLSRPT TXT records with a large TTL (reducing the window for successful application of

transient attacks against DNS resolution of the record) or to deploy DNSSEC on the deploying zone.

8. Privacy Considerations

MTAs are generally considered public knowledge, however, the internals of how those MTAs are configured and the users of those MTAs may not be as public. It should be noted that when providing a receiving site with information, it may reveal information about the sender's configuration, or even information about the senders themselves. Consider that by sending a report, it might disclose your SSL library version as the inability to negotiate a session may be a known incompatbility between two library versions, or perhaps commonly used in a operating system release that is centered in a certain region. The risk may be minimal, but should be considered.

9. References

9.1. Normative References

[I-D.ietf-uta-mta-sts]
Margolis, D., Risher, M., Ramakrishnan, B., Brotman, A., and J. Jones, "SMTP MTA Strict Transport Security (MTA-STS)", draft-ietf-uta-mta-sts-19 (work in progress), May 2018.

[RFC1952] Deutsch, P., "GZIP file format specification version 4.3", RFC 1952, DOI 10.17487/RFC1952, May 1996, <https://www.rfc-editor.org/info/rfc1952>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, DOI 10.17487/RFC3492, March 2003, <https://www.rfc-editor.org/info/rfc3492>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC4408] Wong, M. and W. Schlitt, "Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1", RFC 4408, DOI 10.17487/RFC4408, April 2006, <https://www.rfc-editor.org/info/rfc4408>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5321] Klensin, J., "Simple Mail Transfer Protocol", RFC 5321, DOI 10.17487/RFC5321, October 2008, <https://www.rfc-editor.org/info/rfc5321>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC5891] Klensin, J., "Internationalized Domain Names in Applications (IDNA): Protocol", RFC 5891, DOI 10.17487/RFC5891, August 2010, <https://www.rfc-editor.org/info/rfc5891>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC6068] Duerst, M., Masinter, L., and J. Zawinski, "The 'mailto' URI Scheme", RFC 6068, DOI 10.17487/RFC6068, October 2010, <https://www.rfc-editor.org/info/rfc6068>.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, DOI 10.17487/RFC6125, March 2011, <https://www.rfc-editor.org/info/rfc6125>.

[RFC6376] Crocker, D., Ed., Hansen, T., Ed., and M. Kucherawy, Ed., "DomainKeys Identified Mail (DKIM) Signatures", STD 76, RFC 6376, DOI 10.17487/RFC6376, September 2011, <https://www.rfc-editor.org/info/rfc6376>.

[RFC6522] Kucherawy, M., Ed., "The Multipart/Report Media Type for the Reporting of Mail System Administrative Messages", STD 73, RFC 6522, DOI 10.17487/RFC6522, January 2012, <https://www.rfc-editor.org/info/rfc6522>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <https://www.rfc-editor.org/info/rfc6698>.

[RFC6713] Levine, J., "The 'application/zlib' and 'application/gzip' Media Types", RFC 6713, DOI 10.17487/RFC6713, August 2012, <https://www.rfc-editor.org/info/rfc6713>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/info/rfc7405>.

[RFC7493] Bray, T., Ed., "The I-JSON Message Format", RFC 7493, DOI 10.17487/RFC7493, March 2015, <https://www.rfc-editor.org/info/rfc7493>.

[RFC7672] Dukhovni, V. and W. Hardaker, "SMTP Security via Opportunistic DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS)", RFC 7672, DOI 10.17487/RFC7672, October 2015, <https://www.rfc-editor.org/info/rfc7672>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[RFC3207] Hoffman, P., "SMTP Service Extension for Secure SMTP over Transport Layer Security", RFC 3207, DOI 10.17487/RFC3207, February 2002, <https://www.rfc-editor.org/info/rfc3207>.

[RFC3464] Moore, K. and G. Vaudreuil, "An Extensible Message Format for Delivery Status Notifications", RFC 3464, DOI 10.17487/RFC3464, January 2003, <https://www.rfc-editor.org/info/rfc3464>.

[RFC3501] Crispin, M., "INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1", RFC 3501, DOI 10.17487/RFC3501, March 2003, <https://www.rfc-editor.org/info/rfc3501>.

[RFC3864] Klyne, G., Nottingham, M., and J. Mogul, "Registration Procedures for Message Header Fields", BCP 90, RFC 3864, DOI 10.17487/RFC3864, September 2004, <https://www.rfc-editor.org/info/rfc3864>.

[RFC6533] Hansen, T., Ed., Newman, C., and A. Melnikov, "Internationalized Delivery Status and Disposition Notifications", RFC 6533, DOI 10.17487/RFC6533, February 2012, <https://www.rfc-editor.org/info/rfc6533>.

[RFC7321] McGrew, D. and P. Hoffman, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 7321, DOI 10.17487/RFC7321, August 2014, <https://www.rfc-editor.org/info/rfc7321>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <https://www.rfc-editor.org/info/rfc7435>.

[RFC7469] Evans, C., Palmer, C., and R. Sleevi, "Public Key Pinning Extension for HTTP", RFC 7469, DOI 10.17487/RFC7469, April 2015, <https://www.rfc-editor.org/info/rfc7469>.

[RFC7489] Kucherawy, M., Ed. and E. Zwicky, Ed., "Domain-based Message Authentication, Reporting, and Conformance (DMARC)", RFC 7489, DOI 10.17487/RFC7489, March 2015, <https://www.rfc-editor.org/info/rfc7489>.

[RFC8098] Hansen, T., Ed. and A. Melnikov, Ed., "Message Disposition Notification", STD 85, RFC 8098, DOI 10.17487/RFC8098, February 2017, <https://www.rfc-editor.org/info/rfc8098>.

9.3. URIs

[1] Section 2.2.3

[2] Section 3

Appendix A. Example Reporting Policy

A.1. Report using MAILTO

```
            _smtp._tls.mail.example.com. IN TXT \
                    "v=TLSRPTv1;rua=mailto:reports@example.com"
```

A.2. Report using HTTPS

```
           _smtp._tls.mail.example.com. IN TXT \
                   "v=TLSRPTv1; \
                   rua=https://reporting.example.com/v1/tlsrpt"
```

Appendix B. Example JSON Report

Below is an example JSON report for messages from Company-X to Company-Y, where 100 sessions were attempted to Company Y servers with an expired certificate and 200 sessions were attempted to Company Y servers that did not successfully respond to the "STARTTLS" command. Additionally 3 sessions failed due to "X509_V_ERR_PROXY_PATH_LENGTH_EXCEEDED".

```
{
  "organization-name": "Company-X",
  "date-range": {
    "start-datetime": "2016-04-01T00:00:00Z",
    "end-datetime": "2016-04-01T23:59:59Z"
  },
  "contact-info": "sts-reporting@company-x.example",
  "report-id": "5065427c-23d3-47ca-b6e0-946ea0e8c4be",
  "policies": [{
    "policy": {
      "policy-type": "sts",
      "policy-string": ["version: STSv1","mode: testing",
            "mx: *.mail.company-y.example","max_age: 86400"],
      "policy-domain": "company-y.example",
      "mx-host": "*.mail.company-y.example"
    },
    "summary": {
      "total-successful-session-count": 5326,
      "total-failure-session-count": 303
    },
    "failure-details": [{
      "result-type": "certificate-expired",
      "sending-mta-ip": "2001:db8:abcd:0012::1",
      "receiving-mx-hostname": "mx1.mail.company-y.example",
      "failed-session-count": 100
    }, {
      "result-type": "starttls-not-supported",
      "sending-mta-ip": "2001:db8:abcd:0013::1",
      "receiving-mx-hostname": "mx2.mail.company-y.example",
      "receiving-ip": "203.0.113.56",
      "failed-session-count": 200,
      "additional-information": "https://reports.company-x.example/
        report_info ? id = 5065427 c - 23 d3# StarttlsNotSupported "
    }, {
      "result-type": "validation-failure",
      "sending-mta-ip": "198.51.100.62",
      "receiving-ip": "203.0.113.58",
      "receiving-mx-hostname": "mx-backup.mail.company-y.example",
      "failed-session-count": 3,
      "failure-error-code": "X509_V_ERR_PROXY_PATH_LENGTH_EXCEEDED"
    }]
  }]
}
```

Authors' Addresses

Daniel Margolis
Google, Inc

Email: dmargolis@google.com

Alexander Brotman
Comcast, Inc

Email: alex_brotman@comcast.com

Binu Ramakrishnan
Yahoo!, Inc

Email: rbinu@oath.com

Janet Jones
Microsoft, Inc

Email: janet.jones@microsoft.com

Mark Risher
Google, Inc

Email: risher@google.com