Postfix TLS feature summary in SMTP client delivery status logging

wietse@porcupine.org
September, 2025

[NOTE: The implementation differs from this design. The official documentation now lives at https://www.postfix.org/postconf.5.html#smtp_log_tls_feature_status]

Objective

Make it easy to extract from Postfix logging how email deliveries have used Postfix TLS features such as the TLS security level and upcoming REQUIRETLS support. This will be achieved by adding a number of additional features to Postfix SMTP client delivery status logging. The hasty reader can skip directly to the examples in the "Proposal" and "Implementation" sections.
One obvious candidate feature is the TLS security level ('none', 'may', 'encrypt', or one of the authenticated levels), with an indication of whether or not the desired level was achieved.

There are also a number of TLS-related protocols that could benefit from a success or failure indication in delivery status logging:

MTA-STS (RFC 8461)	Relies on DNS and HTTPS for policy discovery, and uses certificate matching for authentication.
REQUIRETLS (RFC 8689)	Allows a sender to request authenticated TLS for all hops in the forward path of a message.

Non-objectives

No additional logging of TLS handshake details. Currently, Postfix TLS client code logs the result of a successful TLS handshake upon its completion, or when a connection is reused. This logs the TLS protocol name, and the algorithms used for key exchange and for encryption (for an example, see "Appendix: existing TLS handshake logging"). Viktor Dukhovni has indicated that currently-available handshake information is not useful for delivery status logging.

Background

Postfix delivery status logging shows when a delivery is completed, deferred, or abandoned,

together with some forensic information such as time stamps, queue ID, where delivery was attempted to, the delivery status, and a remote server response or local message disposition information. For an example, see "Appendix: existing delivery status logging".

Requirements

The logging must show the status of the following security features:

- The attempted Postfix TLS security level (none, may, encrypt, and so on).
    - This includes the pseudo level 'sts' (a custom variant of 'secure').
- Protocols that depend on TLS, if activated. There is one: REQUIRETLS.

The logging must show that a feature was 'full' or 'relaxed':

- A 'full' feature means that every RFC requirement for that feature must be met.
    - TLS example: the TLS handshake result must satisfy the TLS security level.
    - REQUIRETLS example: the remote server certificate must match, and the server must announce REQUIRETLS support.
- A 'relaxed' feature means that some RFC requirements are relaxed.
    - DANE example: allow DNSSEC-signed TLSA records for MX hosts found with insecure DNS.
    - REQUIRETLS example: relax the requirement for a server certificate match.

The logging must show for each active feature a policy status 'compliant', 'violation' or 'undecided':

- A 'compliant' status means that a 'full' or 'relaxed' feature satisfies policy.
- A 'violation' status means that some policy requirement was not satisfied.
    - TLS example: the TLS handshake result did not satisfy the TLS policy level.
    - REQUIRETLS examples: the TLS security level disables encryption or server authentication; the remote server certificate did not match; or the remote server did not announce REQUIRETLS support.
- An 'undecided' status means that a connection broke before that feature's 'compliant' or 'violation' status could be determined. This information is useful because it shows Postfix's intentions, that is, what could happen when a connection is tried again.
    - Example: a connection needs REQUIRELS, and breaks before the TLS handshake is completed. In that case, log both the TLS level status and REQUIRETLS status as 'undecided'.

Proposal

- In TLS status logging. the TLS level feature name will be 'none', 'may',' encrypt', ..., 'dane', and the REQUIRETLS extension's feature name will be "requiretls".
- Prepend 'disabled:' to the above when a feature is unavailable.
- Enclose the above in "(" and ")"to indicate relaxed enforcement of that feature.

- Prepend to the above one "!" to indicate a 'policy violation' status.
- Append to the above one "?" to indicate a 'policy undecided' status.
- The proposed format of the new TLS status attribute is illustrated with examples below. This format uses the same structure as the existing 'delays' logging: "tag=value/value…".

Scenario	**`Logging`**
A connection that does not use TLS.	`tls=none`
Opportunistic TLS, successful handshake	`tls=may`
Opportunistic TLS, after fallback to plaintext.	`tls=(disabled:may)`
Mandatory TLS policy compliant	`tls=encrypt`
Mandatory TLS policy undecided due to connection failure	`tls=encrypt?`
DANE policy compliant	`tls=dane`
Relaxed DANE policy compliant after insecure MX lookup	`tls=(dane)`
DANE policy undecided due to connection failure	`tls=dane?`
DANE policy compliant after fall-back to 'encrypt'.	`tls=(dane)`
DANE-ONLY policy violation after no certificate match	`tls=!dane-only`
Secure policy compliant	`tls=secure`
Secure policy undecided due to connection failure	`tls=secure?`
Secure policy violation, after no certificate match	`tls=!secure`
Need DANE + REQUIRETLS, failure before/in TLS handshake	`tls=dane?/requiretls?`
DANE policy compliant + REQUIRETLS policy compliant	`tls=dane/requiretls`
Secure policy violation implies REQUIRETLS policy violation	`tls=!secure/!requiretls`
Opportunistic TLS + opportunistic REQUIRETLS, server does not announce REQUIRETLS support	`tls=may/(disabled:requiretls)`
Opportunistic TLS with "relaxed REQUIRETLS" success	`tls=may/(requiretls)`

Implementation

- Before making a connection, reset the status of all TLS-related policy features to 'undecided' when the feature is active.
- The TLS level 'none' policy cannot fail, but it may remain 'undecided' when a connection attempt fails (Postfix was unable to make a connection).
- Other TLS levels:
    - The TLS handshake result determines whether the TLS level policy succeeds or fails. In the case of a TLS level policy failure, also flag the features that depend on TLS (REQUIRETLS, etc.) as a policy failure.
    - Corollary: if a connection fails before or during a TLS handshake, all TLS-related policy features remain 'undecided'.
- As the SMTP-over-TLS engine proceeds, decide if the REQUIRETLS etc. policy succeeds or fails.
- Insert the "`tls=whatever/whatever...`" logging attribute somewhere before the "`dsn=xxx, status=yyy (server response)`" attributes because they cover the same thing and because the last item contains free text. I suggest adding the new attribute between "`delays=`" and "`dsn=`".
- Below is a hypothetical example for delivery that successfully uses opportunistic TLS and no other TLS-related SMTP extension:

  ```
  Aug 24 09:20:50 wzv postfix/smtp[2009999]: 4c8vgk2lXhzNcrX:
  to=<wietse@porcupine.org>, orig_to=<wietse>,
  relay=spike.porcupine.org[168.100.3.2]:25,conn_use=2, delay=0.31,
  delays=0.06/0.05/0.07/0.12, tls=may, dsn=2.0.0, status=sent (250
  2.0.0 Ok: queued as 4c8vgk3xlZzJrNm)
  ```

- Below is another hypothetical example for delivery that successfully uses the REQUIRETLS extension while relaxing the requirement for server certificate matching.

  ```
  Aug 24 09:20:50 wzv postfix/smtp[2009999]: 4c8vgk2lXhzNcrX:
  to=<wietse@porcupine.org>, orig_to=<wietse>,
  relay=spike.porcupine.org[168.100.3.2]:25,conn_use=2, delay=0.31,
  delays=0.06/0.05/0.07/0.12, tls=encrypt/(requiretls), dsn=2.0.0,
  status=sent (250 2.0.0 Ok: queued as 4c8vgk3xlZzJrNm)
  ```

Appendix: existing delivery status logging

Below is a sample of Postfix SMTP client delivery status logging.

```
Aug 24 09:20:50 wzv postfix/smtp[2009999]: 4c8vgk2lXhzNcrX:
to=<wietse@porcupine.org>, orig_to=<wietse>,
relay=spike.porcupine.org[168.100.3.2]:25,conn_use=2, delay=0.31,
delays=0.06/0.05/0.07/0.12, dsn=2.0.0, status=sent (250 2.0.0 Ok:
queued as 4c8vgk3xlZzJrNm)
```

This logging contains

- `Aug 24 09:20:50` The date and time stamp.
- `wzv` the name of the logging host.
- `postfix/smtp[2009999]` the name and ID of the logging process.
- `4c8vgk2lXhzNcrX` The message queue ID: .
- `to=<wietse@porcupine.org>` the final envelope recipient.
- `orig_to=<wietse>` the optional original recipient.
- `relay=spike.porcupine.org[168.100.3.2]:25` the remote SMTP server hostname, IP address, and TCP port.
- `conn_use=2` the optional connection reuse counter.
- `delay=0.31` the time from message arrival to completion of the delivery attempt.
- `delays=0.06/0.05/0.07/0.12`: detailed breakdown of the over-all delay.
    - `0.06` the time from arrival until the scheduler opened the queue file,
    - `0.05` the time before the scheduler contacted a Postfix SMTP client process
    - `0.07` the time in DNS lookup, and in TCP, SMTP and TLS handshakes,
    - `0.12` the time spent delivering the message.
- `dsn=2.0.0` the RFC 3463 delivery status
- `status=sent` the Postfix delivery status, usually "sent", "deferred", or "bounced".
- And text with a server response or other information about the delivery such as "delivered to command", "delivered to file", or "forwarded as <queue id>".

Appendix: existing TLS handshake logging

Currently TLS events are logged separately at the time that the events happen, such as:

```
Aug 24 09:20:50 wzv postfix/smtp[2009999]: Untrusted TLS
connection established to spike.porcupine.org[168.100.3.2]:25:
TLSv1.3 with cipher TLS_AES_256_GCM_SHA384 (256/256 bits)
key-exchange x25519 server-signature RSA-PSS (2048 bits)
server-digest SHA256

Aug 24 11:25:24 wzv postfix/smtp[2036199]: Untrusted TLS
connection reused to spike.porcupine.org[168.100.3.2]:25: TLSv1.3
```

```
with cipher TLS_AES_256_GCM_SHA384 (256/256 bits) key-exchange
x25519 server-signature RSA-PSS (2048 bits) server-digest SHA256
```