

Workgroup: Internet Engineering Task Force  
Internet-Draft:  
draft-ietf-opsawg-tlstm-update-14  
Updates: [6353](#) (if approved)  
Published: 30 March 2023  
Intended Status: Standards Track  
Expires: 1 October 2023  
Authors: K. Vaughn, Ed.  
Trevilon LLC

## Updates to the TLS Transport Model for SNMP

### Abstract

This document updates RFC 6353 "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", to reflect changes necessary to support Transport Layer Security Version 1.3 (TLS 1.3) and Datagram Transport Layer Security Version 1.3 (DTLS 1.3), which are jointly known as "(D)TLS 1.3". This document is compatible with (D)TLS 1.2 and is intended to be compatible with future versions of SNMP and (D)TLS.

This document updates the SNMP-TLS-TM-MIB as defined in RFC 6353.

### 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 1 October 2023.

### Copyright Notice

Copyright (c) 2023 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

## Table of Contents

- [1. Introduction](#)
  - [1.1. Conventions](#)
- [2. Changes from RFC 6353](#)
  - [2.1. TLSTM Fingerprint](#)
  - [2.2. Security Level](#)
  - [2.3. \(D\)TLS Version](#)
- [3. Additional Rules for TLS 1.3](#)
  - [3.1. Zero Round Trip Time Resumption \(0-RTT\)](#)
  - [3.2. TLS cipher suites, extensions and protocol invariants](#)
- [4. MIB Module Definition](#)
- [5. Security Considerations](#)
- [6. IANA Considerations](#)
- [7. Acknowledgements](#)
- [8. References](#)
  - [8.1. Normative References](#)
  - [8.2. Informative References](#)
- [Author's Address](#)

## 1. Introduction

This document updates and clarifies how the rules of [[RFC6353](#)] apply when using Transport Layer Security (TLS) or Datagram Transport Layer Security (DTLS) versions later than 1.2. This document jointly refers to these two protocols as "(D)TLS". The update also emphasizes the [[RFC8996](#)] requirement that prohibits the use of TLS versions prior to TLS 1.2 when using SNMP. Although the text of this document specifically references SNMPv3 and (D)TLS 1.3, this document may be applicable to future versions of these protocols and is backwards compatible with (D)TLS 1.2.

### 1.1. Conventions

Within this document the terms "TLS", "DTLS", and "(D)TLS" apply to all versions of the indicated protocols. The term "SNMP" means "SNMPv3" unless a specific version number is indicated. Specific version numbers are used when the text needs to emphasize version numbers.

For consistency with SNMP-related specifications, this document favors terminology as defined in [[STD62](#)], rather than favoring

terminology that is consistent with non-SNMP specifications. This is consistent with the IESG decision to not require the SNMP terminology be modified to match the usage of other non-SNMP specifications when SNMP was advanced to a Full Standard.

"Authentication" in this document typically refers to the English meaning of "serving to prove the authenticity of" the message, not data source authentication or peer identity authentication. The terms "manager" and "agent" are not used in this document because, in the [RFC3411](#) architecture, all SNMP entities have the capability of acting as manager, agent, or both depending on the SNMP application types supported in the implementation. Where distinction is necessary, the application names of command generator, command responder, notification originator, notification receiver, and proxy forwarder are used. See "[SNMP Applications](#)" ([RFC3411](#)) for further information.

Throughout this document, the terms "client" and "server" are used to refer to the two ends of the TLS transport connection. The client actively opens the TLS connection, and the server passively listens for the incoming TLS connection. An SNMP entity **MAY** act as a TLS client or server or both, depending on the SNMP applications supported.

Throughout this document, the term "session" is used to refer to a secure association between two instances of the TLS Transport Model (TLSTM) that permits the transmission of one or more SNMP messages within the lifetime of the session. The TLS protocol also has an internal notion of a session and although these two concepts of a session are related, when the term "session" is used this document is referring to the TLSTM's specific session and not directly to the TLS protocol's session.

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.

## 2. Changes from RFC 6353

This document updates [[RFC6353](#)]. The changes from [[RFC6353](#)] are defined in the following clauses.

### 2.1. TLSTM Fingerprint

[[RFC6353](#)] defines a fingerprint algorithm that references the one-octet TLS 1.2 hash algorithm identifier. TLS 1.3 replaced the one-octet hash algorithm identifier with a two-octet TLS 1.3 cipher suite identifier. The TLS community does not plan to ever add

additional values to the TLS 1.2 hash algorithm registry because some might incorrectly infer that using a new hash algorithm with TLS 1.2 would overcome the limitations of TLS 1.2. However, there is still a need within TLSTM to support new values as they are developed.

This document updates the definition of SnmpTLSTLFingerprint to clarify that the one-octet identifier in the fingerprint algorithm uses the IANA SNMP-TLSTM HashAlgorithm Registry; this registry is consistent with the IANA TLS HashAlgorithm Registry for its initial values but can be extended as needed to support new hashing algorithms without implying that the new values can be used by TLS version 1.2. This change allows the reuse of the existing fingerprint TEXTUAL-CONVENTION and minimizes the impact to [\[RFC6353\]](#).

A "Y" in the "Recommended" column indicates that the registered value has been recommended through a formal Standards Action. Not all parameters defined in Standards Track documents are necessarily marked as "Recommended".

An "N" in the "Recommended" column does not necessarily mean that it is flawed; rather, it indicates that the item either has not been through the IETF consensus process, has limited applicability, or is intended only for specific use cases.

The initial values for the SNMP-TLSTM HashAlgorithm Registry are defined below:

Value	Description	Recommended	Reference
0	none	N	[RFC5246]
1	md5	N	[RFC5246]
2	sha1	N	[RFC5246]
3	sha224	Y	[RFC5246]
4	sha256	Y	[RFC5246]
5	sha384	Y	[RFC5246]
6	sha512	Y	[RFC5246]
7	reserved		[RFC8447]
8	intrinsic	N	[RFC8422]
9-223	reserved		[RFC8447]
224-255	private		[RFC5246]

Table 1: SNMP-TLSTM Hash Algorithms

Values 0 through 2 **MUST NOT** be used by implementations of this document but are listed for historical consistency.

## 2.2. Security Level

The [RFC3411](#) architecture recognizes three levels of security:

- \*without authentication and without privacy (noAuthNoPriv)
- \*with authentication but without privacy (authNoPriv)
- \*with authentication and with privacy (authPriv)

Cipher suites for (D)TLS 1.3 defined in [\[RFC8446\]](#) provide both authentication and privacy. Cipher suites defined in [\[RFC9150\]](#) for (D)TLS 1.3 provide only authentication, without any privacy protection. Implementations MAY choose to force (D)TLS 1.3 to only allow cipher suites that provide both authentication and privacy.

## 2.3. (D)TLS Version

[\[RFC6353\]](#) states that TLSTM clients and servers **MUST NOT** request, offer, or use SSL 2.0. [\[RFC8996\]](#) prohibits the use of (D)TLS versions prior to version 1.2. TLSTM **MUST** only be used with (D)TLS version 1.2 and later.

## 3. Additional Rules for TLS 1.3

This document specifies additional rules and clarifications for the use of TLS 1.3. These rules may additionally apply to future versions of TLS.

### 3.1. Zero Round Trip Time Resumption (0-RTT)

TLS 1.3 implementations for SNMP **MUST NOT** enable the 0-RTT mode of session resumption (either sending or accepting) and **MUST NOT** automatically resend 0-RTT data if it is rejected by the server. The reason 0-RTT is disallowed is that there are no "safe" SNMP messages that if replayed will be guaranteed to cause no harm at a server side: all incoming notification or command responses are meant to be acted upon only once. See Security considerations section for further details.

TLS TM clients and servers **MUST NOT** request, offer, or use the 0-RTT mode of TLS 1.3. [\[RFC8446\]](#) removed the renegotiation supported in TLS 1.2 [\[RFC5246\]](#); for session resumption, it introduced a zero-RTT (0-RTT) mode, saving a round-trip at connection setup at the cost of increased risk of replay attacks (it is possible for servers to guard against this attack by keeping track of all the messages received). [\[RFC8446\]](#) requires a profile be written for any application that wants to use 0-RTT, specifying which messages are "safe to use" on this mode. Within SNMP, there are no messages that are "safe to use" with this mode.

Renegotiation of sessions is not supported as it is not supported by TLS 1.3. If a future version of TLS supports renegotiation, this RFC should be updated to indicate whether there are any additional requirements related to its use.

### **3.2. TLS cipher suites, extensions and protocol invariants**

[[RFC8446](#)] section 9 requires that, in the absence of application profiles, certain cipher suites, TLS extensions, and TLS protocol invariants are mandatory to implement. This document does not specify an application profile, hence all the compliance requirements in [[RFC8446](#)] apply.

### **4. MIB Module Definition**

This module makes references to [[RFC1123](#)], [[RFC2578](#)], [[RFC2579](#)], [[RFC2580](#)], [[RFC3411](#)], [[RFC3413](#)], [[RFC5246](#)], [[RFC5280](#)], [[RFC5890](#)], [[RFC5952](#)], [[RFC5953](#)], [[RFC6353](#)], and [[STD58](#)]

SNMP-TLS-TM-MIB DEFINITIONS ::= BEGIN

IMPORTS

```
MODULE-IDENTITY, OBJECT-TYPE,
OBJECT-IDENTITY, mib-2, snmpDomains,
Counter32, Unsigned32, Gauge32, NOTIFICATION-TYPE
  FROM SNMPv2-SMI          -- RFC 2578 or any update thereof
TEXTUAL-CONVENTION, TimeStamp, RowStatus, StorageType,
AutonomousType
  FROM SNMPv2-TC          -- RFC 2579 or any update thereof
MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
  FROM SNMPv2-CONF        -- RFC 2580 or any update thereof
SnmpAdminString
  FROM SNMP-FRAMEWORK-MIB  -- RFC 3411 or any update thereof
snmpTargetParamsName, snmpTargetAddrName
  FROM SNMP-TARGET-MIB    -- RFC 3413 or any update thereof
;
```

snmpTlstmMIB MODULE-IDENTITY

LAST-UPDATED "202303010000Z"

ORGANIZATION "OPSA Working Group"

CONTACT-INFO "WG-EMail: opsawg@ietf.org

Mailing list subscription info:

<https://www.ietf.org/mailman/listinfo/opsawg>

Kenneth Vaughn

Trevilon LLC

1060 Hwy 107 South

Del Rio, TN 37727

United States

Phone: +1 571 331 5670

Email: kvaughn@trevilon.com"

DESCRIPTION "

The TLS Transport Model MIB

Copyright (c) 2010-2022 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Revised BSD License set forth in Section 4.c of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

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 (RFC 2119) (RFC 8174) when, and only when, they appear in all capitals, as shown here."

REVISION "202303010000Z"  
DESCRIPTION "This version of this MIB module is part of RFC XXXX; see the RFC itself for full legal notices. This version: [ Note to RFC Editor: please replace the above XXXX with the RFC number of this document ]  
1. Updates the definition of SnmpTLSFingerprint to clarify the registry used for the one-octet hash algorithm identifier.  
2. Capitalizes key words in conformance with BCP 14  
3. Replaces 'may not' with 'MUST NOT' to clarify intent in several locations.  
4. Replaces 'may not' with a clarification within the definition of SnmpTLSAddress"

REVISION "201107190000Z"  
DESCRIPTION "This version of this MIB module is part of RFC 6353; see the RFC itself for full legal notices. The only change was to introduce new wording to reflect require changes for IDNA addresses in the SnmpTLSAddress TC."

REVISION "201005070000Z"  
DESCRIPTION "This version of this MIB module is part of RFC 5953; see the RFC itself for full legal notices."

::= { mib-2 198 }

-- \*\*\*\*\*  
-- subtrees of the SNMP-TLS-TM-MIB  
-- \*\*\*\*\*

snmpTlstmNotifications OBJECT IDENTIFIER ::= { snmpTlstmMIB 0 }  
snmpTlstmIdentities OBJECT IDENTIFIER ::= { snmpTlstmMIB 1 }  
snmpTlstmObjects OBJECT IDENTIFIER ::= { snmpTlstmMIB 2 }  
snmpTlstmConformance OBJECT IDENTIFIER ::= { snmpTlstmMIB 3 }  
snmpTlstmHashAlgorithms OBJECT-IDENTITY  
STATUS current  
DESCRIPTION  
"A node used to register hashing algorithm identifiers recorded in the IANA SNMP-TLSTM HashAlgorithm Registry."  
::= { snmpTlstmMIB 4 }



```
-- *****  
-- snmpTlstmObjects - Objects  
-- *****
```

snmpTLSTCPDomain OBJECT-IDENTITY

STATUS current

DESCRIPTION

"The SNMP over TLS via TCP transport domain. The corresponding transport address is of type SnmpTLSAddress.

The securityName prefix to be associated with the snmpTLSTCPDomain is 'tls'. This prefix MAY be used by security models or other components to identify which secure transport infrastructure authenticated a securityName."

REFERENCE

"RFC 2579: Textual Conventions for SMIV2"  
 ::= { snmpDomains 8 }

snmpDTLSUDPDDomain OBJECT-IDENTITY

STATUS current

DESCRIPTION

"The SNMP over DTLS via UDP transport domain. The corresponding transport address is of type SnmpTLSAddress.

The securityName prefix to be associated with the snmpDTLSUDPDDomain is 'dtls'. This prefix MAY be used by security models or other components to identify which secure transport infrastructure authenticated a securityName."

REFERENCE

"RFC 2579: Textual Conventions for SMIV2"  
 ::= { snmpDomains 9 }

SnmpTLSAddress ::= TEXTUAL-CONVENTION

DISPLAY-HINT "1a"

STATUS current

DESCRIPTION

"Represents an IPv4 address, an IPv6 address, or a US-ASCII-encoded hostname and port number.

An IPv4 address MUST be in dotted decimal format followed by a colon ':' (US-ASCII character 0x3A) and a decimal port number in US-ASCII.

An IPv6 address MUST be a colon-separated format (as described in RFC 5952), surrounded by square brackets ('[', US-ASCII character 0x5B, and ']', US-ASCII character 0x5D), followed by a colon ':' (US-ASCII character 0x3A) and a decimal port number in US-ASCII.

A hostname MUST be in US-ASCII (as per RFC 1123);

internationalized hostnames MUST be encoded as A-labels as specified in RFC 5890. The hostname is followed by a colon ':' (US-ASCII character 0x3A) and a decimal port number in US-ASCII. The name SHOULD be fully qualified whenever possible.

Values of this textual convention are not guaranteed to be directly usable as transport layer addressing information, potentially requiring additional processing, such as run-time resolution. As such, applications that write them MUST be prepared for handling errors if such values are not supported, or cannot be resolved (if resolution occurs at the time of the management operation).

The DESCRIPTION clause of TransportAddress objects that may have SnmpTLSAddress values MUST fully describe how (and when) such names are to be resolved to IP addresses and vice versa.

This textual convention SHOULD NOT be used directly in object definitions since it restricts addresses to a specific format. However, if it is used, it MAY be used either on its own or in conjunction with TransportAddressType or TransportDomain as a pair.

When this textual convention is used as a syntax of an index object, there may be issues with the limit of 128 sub-identifiers specified in SMIV2 (STD 58). It is RECOMMENDED that all MIB documents using this textual convention make explicit any limitations on index component lengths that management software MUST observe. This MAY be done either by including SIZE constraints on the index components or by specifying applicable constraints in the conceptual row DESCRIPTION clause or in the surrounding documentation."

#### REFERENCE

- "RFC 1123: Requirements for Internet Hosts - Application and Support
- RFC 5890: Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework
- RFC 5952: A Recommendation for IPv6 Address Text Representation"

SYNTAX OCTET STRING (SIZE (1..255))

SnmpTLSPFingerprint ::= TEXTUAL-CONVENTION

DISPLAY-HINT "1x:1x"

STATUS current

#### DESCRIPTION

"A fingerprint value that can be used to uniquely reference

other data of potentially arbitrary length.

An SnmpTLSFingerprint value is composed of a 1-octet hashing algorithm identifier followed by the fingerprint value. The 1-octet identifier value encoded is taken from IANA SNMP-TLSTM HashAlgorithm Registry. The remaining octets of the SnmpTLSFingerprint value are filled using the results of the hashing algorithm.

Historically, the 1-octet hashing algorithm identifier was based on the IANA TLS HashAlgorithm Registry (RFC 5246); however, this registry is no longer in use for TLS 1.3 and above and are not expected to have any new registrations added to it. To allow the fingerprint algorithm to support additional hashing algorithms that might be used by later versions of (D)TLS, the octet value encoded is now taken from IANA SNMP-TLSTM HashAlgorithm Registry. The initial values within this registry are identical to the values in the TLS HashAlgorithm registry but can be extended to support new hashing algorithms as needed.

This TEXTUAL-CONVENTION allows for a zero-length (blank) SnmpTLSFingerprint value for use in tables where the fingerprint value MAY be optional. MIB definitions or implementations MAY refuse to accept a zero-length value as appropriate."

REFERENCE "<https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml>"

SYNTAX OCTET STRING (SIZE (0..255))

-- Identities for use in the snmpTlstmCertToTSNTable

snmpTlstmCertToTSNMIdentities OBJECT IDENTIFIER

::= { snmpTlstmIdentities 1 }

snmpTlstmCertSpecified OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Directly specifies the tmSecurityName to be used for this certificate. The value of the tmSecurityName to use is specified in the snmpTlstmCertToTSNData column. The snmpTlstmCertToTSNData column MUST contain a non-zero length SnmpAdminString compliant value or the mapping described in this row MUST be considered a failure."

::= { snmpTlstmCertToTSNMIdentities 1 }

snmpTlstmCertSANRFC822Name OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Maps a subjectAltName's rfc822Name to a tmSecurityName. The local part of the rfc822Name is passed unaltered but the host-part of the name MUST be passed in lowercase. This mapping results in a 1:1 correspondence between equivalent subjectAltName rfc822Name values and tmSecurityName values except that the host-part of the name MUST be passed in lowercase.

Example rfc822Name Field: FooBar@Example.COM is mapped to tmSecurityName: FooBar@example.com."

::= { snmpTlstmCertToTSNMIdentities 2 }

snmpTlstmCertSANDNSName OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Maps a subjectAltName's dNSName to a tmSecurityName after first converting it to all lowercase (RFC 5280 does not specify converting to lowercase, so this involves an extra step). This mapping results in a 1:1 correspondence between subjectAltName dNSName values and the tmSecurityName values."

REFERENCE "RFC 5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile."

::= { snmpTlstmCertToTSNMIdentities 3 }

snmpTlstmCertSANIpAddress OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Maps a subjectAltName's iPAddress to a tmSecurityName by transforming the binary encoded address as follows:

1) for IPv4, the value is converted into a decimal-dotted quad address (e.g., '192.0.2.1').

2) for IPv6 addresses, the value is converted into a 32-character all lowercase hexadecimal string without any colon separators.

This mapping results in a 1:1 correspondence between subjectAltName iPAddress values and the tmSecurityName values.

The resulting length of an encoded IPv6 address is the maximum length supported by the View-Based Access Control Model (VACM). Using both the Transport Security Model's support for transport prefixes (see the SNMP-TSM-MIB's snmpTsmConfigurationUsePrefix object for details) will result in securityName lengths that exceed what VACM can handle."

::= { snmpTlstmCertToTSNMIdentities 4 }

snmpTlstmCertSANAny OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Maps any of the following fields using the corresponding mapping algorithms:

Type	Algorithm
rfc822Name	snmpTlstmCertSANRFC822Name
dnsName	snmpTlstmCertSANDNSName
ipAddress	snmpTlstmCertSANIpAddress

The first matching subjectAltName value found in the certificate of the above types MUST be used when deriving the tmSecurityName. The mapping algorithm specified in the 'Algorithm' column MUST be used to derive the tmSecurityName.

This mapping results in a 1:1 correspondence between subjectAltName values and tmSecurityName values. The three sub-mapping algorithms produced by this combined algorithm cannot produce conflicting results between themselves."

::= { snmpTlstmCertToTSNMIidentities 5 }

snmpTlstmCertCommonName OBJECT-IDENTITY

STATUS current

DESCRIPTION

"Maps a certificate's CommonName to a tmSecurityName after converting it to a UTF-8 encoding. The usage of CommonNames is deprecated and users are encouraged to use subjectAltName mapping methods instead. This mapping results in a 1:1 correspondence between certificate CommonName values and tmSecurityName values."

::= { snmpTlstmCertToTSNMIidentities 6 }

-- The snmpTlstmSession Group

snmpTlstmSession OBJECT IDENTIFIER ::= { snmpTlstmObjects 1 }

snmpTlstmSessionOpens OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an openSession() request has been executed as a (D)TLS client, regardless of whether it succeeded or failed."

::= { snmpTlstmSession 1 }

snmpTlstmSessionClientCloses OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times a closeSession() request has been executed as a (D)TLS client, regardless of whether it succeeded or failed."

::= { snmpTlstmSession 2 }

snmpTlstmSessionOpenErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an openSession() request failed to open a session as a (D)TLS client, for any reason."

::= { snmpTlstmSession 3 }

snmpTlstmSessionAccepts OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times a (D)TLS server has accepted a new connection from a client and has received at least one SNMP message through it."

::= { snmpTlstmSession 4 }

snmpTlstmSessionServerCloses OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times a closeSession() request has been executed as a (D)TLS server, regardless of whether it succeeded or failed."

::= { snmpTlstmSession 5 }

snmpTlstmSessionNoSessions OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an outgoing message was dropped because the session associated with the passed tmStateReference was no longer (or was never) available."

::= { snmpTlstmSession 6 }

snmpTlstmSessionInvalidClientCertificates OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an incoming session was not established on a (D)TLS server because the presented client certificate was invalid. Reasons for invalidation include, but are not limited to, cryptographic validation failures or lack of a suitable mapping row in the snmpTlstmCertToTSNTable."

::= { snmpTlstmSession 7 }

snmpTlstmSessionUnknownServerCertificate OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an outgoing session was not established on a (D)TLS client because the server certificate presented by an SNMP over (D)TLS server was invalid because no configured fingerprint or Certification Authority (CA) was acceptable to validate it.

This may result because there was no entry in the snmpTlstmAddrTable or because no path could be found to a known CA."

::= { snmpTlstmSession 8 }

snmpTlstmSessionInvalidServerCertificates OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of times an outgoing session was not established on a (D)TLS client because the server certificate presented by an SNMP over (D)TLS server could not be validated even if the fingerprint or expected validation path was known. That is, a cryptographic validation error occurred during certificate validation processing.

Reasons for invalidation include, but are not limited to, cryptographic validation failures."

::= { snmpTlstmSession 9 }

snmpTlstmSessionInvalidCaches OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of outgoing messages dropped because the

```

        tmStateReference referred to an invalid cache."
    ::= { snmpTlstmSession 10 }

-- Configuration Objects

snmpTlstmConfig          OBJECT IDENTIFIER ::= { snmpTlstmObjects 2 }

-- Certificate mapping

snmpTlstmCertificateMapping OBJECT IDENTIFIER ::= {snmpTlstmConfig 1}

snmpTlstmCertToTSNCount OBJECT-TYPE
    SYNTAX      Gauge32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A count of the number of entries in the
        snmpTlstmCertToTSNTable."
    ::= { snmpTlstmCertificateMapping 1 }

snmpTlstmCertToTSNTableLastChanged OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The value of sysUpTime.0 when the snmpTlstmCertToTSNTable
        was last modified through any means, or 0 if it has not been
        modified since the command responder was started."
    ::= { snmpTlstmCertificateMapping 2 }

snmpTlstmCertToTSNTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SnmpTlstmCertToTSNEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table is used by a (D)TLS server to map the (D)TLS
        client's presented X.509 certificate to a tmSecurityName.

        On an incoming (D)TLS/SNMP connection, the client's presented
        certificate MUST either be validated based on an established
        trust anchor, or it MUST directly match a fingerprint in this
        table.  This table does not provide any mechanisms for
        configuring the trust anchors; the transfer of any needed
        trusted certificates for path validation is expected to occur
        through an out-of-band transfer.

        Once the certificate has been found acceptable (either by
        path validation or directly matching a fingerprint in this
        table), this table is consulted to determine the appropriate
        tmSecurityName to identify with the remote connection.  This

```



is done by considering each active row from this table in prioritized order according to its `snmpTlstmCertToTSNID` value. Each row's `snmpTlstmCertToTSNFingerprint` value determines whether the row is a match for the incoming connection:

- 1) If the row's `snmpTlstmCertToTSNFingerprint` value identifies the presented certificate, then consider the row as a successful match.
- 2) If the row's `snmpTlstmCertToTSNFingerprint` value identifies a locally held copy of a trusted CA certificate and that CA certificate was used to validate the path to the presented certificate, then consider the row as a successful match.

Once a matching row has been found, the `snmpTlstmCertToTSNMapType` value can be used to determine how the `tmSecurityName` to associate with the session should be determined. See the `snmpTlstmCertToTSNMapType` column's DESCRIPTION for details on determining the `tmSecurityName` value. If it is impossible to determine a `tmSecurityName` from the row's data combined with the data presented in the certificate, then additional rows MUST be searched looking for another potential match. If a resulting `tmSecurityName` mapped from a given row is not compatible with the needed requirements of a `tmSecurityName` (e.g., VACM imposes a 32-octet-maximum length and the certificate derived `securityName` could be longer), then it MUST be considered an invalid match and additional rows MUST be searched looking for another potential match.

If no matching and valid row can be found, the connection MUST be closed and SNMP messages MUST NOT be accepted over it.

Missing values of `snmpTlstmCertToTSNID` are acceptable and implementations SHOULD continue to the next highest numbered row. It is RECOMMENDED that administrators skip index values to leave room for the insertion of future rows (for example, use values of 10 and 20 when creating initial rows).

Users are encouraged to make use of certificates with `subjectAltName` fields that can be used as `tmSecurityNames` so that a single root CA certificate can allow all child certificate's `subjectAltName` to map directly to a `tmSecurityName` via a 1:1 transformation. However, this table is flexible to allow for situations where existing deployed certificate infrastructures do not provide adequate

subjectAltName values for use as tmSecurityNames. Certificates MAY also be mapped to tmSecurityNames using the CommonName portion of the Subject field. However, the usage of the CommonName field is deprecated and thus this usage is NOT RECOMMENDED. Direct mapping from each individual certificate fingerprint to a tmSecurityName is also possible but requires one entry in the table per tmSecurityName and requires more management operations to completely configure a device."

```
::= { snmpTlstmCertificateMapping 3 }
```

snmpTlstmCertToTSNEntry OBJECT-TYPE

SYNTAX SnmpTlstmCertToTSNEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A row in the snmpTlstmCertToTSNTable that specifies a mapping for an incoming (D)TLS certificate to a tmSecurityName to use for a connection."

INDEX { snmpTlstmCertToTSNID }

```
::= { snmpTlstmCertToTSNTable 1 }
```

SnmpTlstmCertToTSNEntry ::= SEQUENCE {

snmpTlstmCertToTSNID Unsigned32,

snmpTlstmCertToTSNFingerprint SnmpTLSFingerprint,

snmpTlstmCertToTSNMapType AutonomousType,

snmpTlstmCertToTSNData OCTET STRING,

snmpTlstmCertToTSNStorageType StorageType,

snmpTlstmCertToTSNRowStatus RowStatus

}

snmpTlstmCertToTSNID OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A unique, prioritized index for the given entry. Lower numbers indicate a higher priority."

```
::= { snmpTlstmCertToTSNEntry 1 }
```

snmpTlstmCertToTSNFingerprint OBJECT-TYPE

SYNTAX SnmpTLSFingerprint (SIZE(1..255))

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A cryptographic hash of an X.509 certificate. The results of a successful matching fingerprint to either the trusted CA in the certificate validation path or to the certificate itself is dictated by the snmpTlstmCertToTSNMapType column."

```
::= { snmpTlstmCertToTSNEntry 2 }
```

```
snmpTlstmCertToTSNMapType OBJECT-TYPE
```

```
SYNTAX      AutonomousType
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"Specifies the mapping type for deriving a tmSecurityName from a certificate. Details for mapping of a particular type SHALL be specified in the DESCRIPTION clause of the OBJECT-IDENTITY that describes the mapping. If a mapping succeeds it will return a tmSecurityName for use by the TLSTM model and processing stops.

If the resulting mapped value is not compatible with the needed requirements of a tmSecurityName (e.g., VACM imposes a 32-octet-maximum length and the certificate derived securityName could be longer), then future rows MUST be searched for additional snmpTlstmCertToTSNFingerprint matches to look for a mapping that succeeds.

Suitable values for assigning to this object that are defined within the SNMP-TLS-TM-MIB can be found in the snmpTlstmCertToTSNMIdentities portion of the MIB tree."

```
DEFVAL { snmpTlstmCertSpecified }
```

```
::= { snmpTlstmCertToTSNEntry 3 }
```

```
snmpTlstmCertToTSNData OBJECT-TYPE
```

```
SYNTAX      OCTET STRING (SIZE(0..1024))
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"Auxiliary data used as optional configuration information for a given mapping specified by the snmpTlstmCertToTSNMapType column. Only some mapping systems will make use of this column. The value in this column MUST be ignored for any mapping type that does not require data present in this column."

```
DEFVAL { "" }
```

```
::= { snmpTlstmCertToTSNEntry 4 }
```

```
snmpTlstmCertToTSNStorageType OBJECT-TYPE
```

```
SYNTAX      StorageType
```

```
MAX-ACCESS  read-create
```

```
STATUS      current
```

```
DESCRIPTION
```

"The storage type for this conceptual row. Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row."

```

DEFVAL      { nonVolatile }
::= { snmpTlstmCertToTSNEntry 5 }

snmpTlstmCertToTSNRowStatus OBJECT-TYPE
SYNTAX      RowStatus
MAX-ACCESS  read-create
STATUS      current
DESCRIPTION
    "The status of this conceptual row.  This object MAY be used
    to create or remove rows from this table.

    To create a row in this table, an administrator MUST set this
    object to either createAndGo(4) or createAndWait(5).

    Until instances of all corresponding columns are
    appropriately configured, the value of the corresponding
    instance of the snmpTlstmParamsRowStatus column is
    notReady(3).

    In particular, a newly created row cannot be made active
    until the corresponding snmpTlstmCertToTSNFingerprint,
    snmpTlstmCertToTSNMapType, and snmpTlstmCertToTSNData columns
    have been set.

    The following objects MUST NOT be modified while the
    value of this object is active(1):
        - snmpTlstmCertToTSNFingerprint
        - snmpTlstmCertToTSNMapType
        - snmpTlstmCertToTSNData
    An attempt to set these objects while the value of
    snmpTlstmParamsRowStatus is active(1) will result in
    an inconsistentValue error."
::= { snmpTlstmCertToTSNEntry 6 }

-- Maps tmSecurityNames to certificates for use by the
-- SNMP-TARGET-MIB

snmpTlstmParamsCount OBJECT-TYPE
SYNTAX      Gauge32
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "A count of the number of entries in the
    snmpTlstmParamsTable."
::= { snmpTlstmCertificateMapping 4 }

snmpTlstmParamsTableLastChanged OBJECT-TYPE
SYNTAX      TimeStamp
MAX-ACCESS  read-only
STATUS      current

```

DESCRIPTION

"The value of sysUpTime.0 when the snmpTlstmParamsTable was last modified through any means, or 0 if it has not been modified since the command responder was started."

::= { snmpTlstmCertificateMapping 5 }

snmpTlstmParamsTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpTlstmParamsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table is used by a (D)TLS client when a (D)TLS connection is being set up using an entry in the SNMP-TARGET-MIB. It extends the SNMP-TARGET-MIB's snmpTargetParamsTable with a fingerprint of a certificate to use when establishing such a (D)TLS connection."

::= { snmpTlstmCertificateMapping 6 }

snmpTlstmParamsEntry OBJECT-TYPE

SYNTAX SnmpTlstmParamsEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A conceptual row containing a fingerprint hash of a locally held certificate for a given snmpTargetParamsEntry. The values in this row SHOULD be ignored if the connection that needs to be established, as indicated by the SNMP-TARGET-MIB infrastructure, is not a certificate and (D)TLS based connection. The connection SHOULD NOT be established if the certificate fingerprint stored in this entry does not point to a valid locally held certificate or if it points to an unusable certificate (such as might happen when the certificate's expiration date has been reached)."

INDEX { IMPLIED snmpTargetParamsName }

::= { snmpTlstmParamsTable 1 }

```
SnmpTlstmParamsEntry ::= SEQUENCE {
    snmpTlstmParamsClientFingerprint SnmpTLSEFingerprint,
    snmpTlstmParamsStorageType       StorageType,
    snmpTlstmParamsRowStatus         RowStatus
}
```

snmpTlstmParamsClientFingerprint OBJECT-TYPE

SYNTAX SnmpTLSEFingerprint

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object stores the hash of the public portion of a locally held X.509 certificate. The X.509 certificate, its

public key, and the corresponding private key will be used when initiating a (D)TLS connection as a (D)TLS client."  
 ::= { snmpTlstmParamsEntry 1 }

snmpTlstmParamsStorageType OBJECT-TYPE

SYNTAX StorageType  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION  
 "The storage type for this conceptual row. Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row."  
DEFVAL { nonVolatile }  
 ::= { snmpTlstmParamsEntry 2 }

snmpTlstmParamsRowStatus OBJECT-TYPE

SYNTAX RowStatus  
MAX-ACCESS read-create  
STATUS current  
DESCRIPTION  
 "The status of this conceptual row. This object MAY be used to create or remove rows from this table.  
  
To create a row in this table, an administrator MUST set this object to either createAndGo(4) or createAndWait(5).  
  
Until instances of all corresponding columns are appropriately configured, the value of the corresponding instance of the snmpTlstmParamsRowStatus column is notReady(3).  
  
In particular, a newly created row cannot be made active until the corresponding snmpTlstmParamsClientFingerprint column has been set.  
  
The snmpTlstmParamsClientFingerprint object MUST NOT be modified while the value of this object is active(1).  
  
An attempt to set these objects while the value of snmpTlstmParamsRowStatus is active(1) will result in an inconsistentValue error."  
 ::= { snmpTlstmParamsEntry 3 }

snmpTlstmAddrCount OBJECT-TYPE

SYNTAX Gauge32  
MAX-ACCESS read-only  
STATUS current  
DESCRIPTION  
 "A count of the number of entries in the snmpTlstmAddrTable."  
 ::= { snmpTlstmCertificateMapping 7 }

snmpTlstmAddrTableLastChanged OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of sysUpTime.0 when the snmpTlstmAddrTable was last modified through any means, or 0 if it has not been modified since the command responder was started."

::= { snmpTlstmCertificateMapping 8 }

snmpTlstmAddrTable OBJECT-TYPE

SYNTAX SEQUENCE OF SnmpTlstmAddrEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table is used by a (D)TLS client when a (D)TLS connection is being set up using an entry in the SNMP-TARGET-MIB. It extends the SNMP-TARGET-MIB's

snmpTargetAddrTable so that the client can verify that the correct server has been reached. This verification can use either a certificate fingerprint, or an identity authenticated via certification path validation.

If there is an active row in this table corresponding to the entry in the SNMP-TARGET-MIB that was used to establish the connection, and the row's snmpTlstmAddrServerFingerprint column has non-empty value, then the server's presented certificate is compared with the snmpTlstmAddrServerFingerprint value (and the snmpTlstmAddrServerIdentity column is ignored). If the fingerprint matches, the verification has succeeded. If the fingerprint does not match, then the connection MUST be closed.

If the server's presented certificate has passed certification path validation [RFC5280] to a configured trust anchor, and an active row exists with a zero-length snmpTlstmAddrServerFingerprint value, then the snmpTlstmAddrServerIdentity column contains the expected host name. This expected host name is then compared against the server's certificate as follows:

- Implementations MUST support matching the expected host name against a dNSName in the subjectAltName extension field and MAY support checking the name against the CommonName portion of the subject distinguished name.

- The '\*' (ASCII 0x2a) wildcard character is allowed in the dNSName of the subjectAltName extension (and in common name, if used to store the host name), but only as the left-most (least significant) DNS label in that value. This wildcard matches any left-most DNS label in the server name. That is, the subject \*.example.com matches the server names a.example.com and b.example.com, but does not match example.com or a.b.example.com. Implementations MUST support wildcards in certificates as specified above, but MAY provide a configuration option to disable them.
- If the locally configured name is an internationalized domain name, conforming implementations MUST convert it to the ASCII Compatible Encoding (ACE) format for performing comparisons, as specified in Section 7 of [RFC5280].

If the expected host name fails these conditions then the connection MUST be closed.

If there is no row in this table corresponding to the entry in the SNMP-TARGET-MIB and the server can be authorized by another, implementation-dependent means, then the connection MAY still proceed."

```
 ::= { snmpTlstmCertificateMapping 9 }
```

```
snmpTlstmAddrEntry OBJECT-TYPE
```

```
SYNTAX      SnmpTlstmAddrEntry
```

```
MAX-ACCESS  not-accessible
```

```
STATUS      current
```

```
DESCRIPTION
```

"A conceptual row containing a copy of a certificate's fingerprint for a given snmpTargetAddrEntry. The values in this row SHOULD be ignored if the connection that needs to be established, as indicated by the SNMP-TARGET-MIB infrastructure, is not a (D)TLS based connection. If an snmpTlstmAddrEntry exists for a given snmpTargetAddrEntry, then the presented server certificate MUST match or the connection MUST NOT be established. If a row in this table does not exist to match an snmpTargetAddrEntry row, then the connection SHOULD still proceed if some other certificate validation path algorithm (e.g., RFC 5280) can be used."

```
INDEX      { IMPLIED snmpTargetAddrName }
```

```
 ::= { snmpTlstmAddrTable 1 }
```

```
SnmpTlstmAddrEntry ::= SEQUENCE {
```

```
  snmpTlstmAddrServerFingerprint      SnmpTLSEFingerprint,
```

```
  snmpTlstmAddrServerIdentity          SnmpAdminString,
```

```
  snmpTlstmAddrStorageType             StorageType,
```

```
  snmpTlstmAddrRowStatus               RowStatus
```



}

snmpTlstmAddrServerFingerprint OBJECT-TYPE

SYNTAX SnmpTLSEFingerprint

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A cryptographic hash of a public X.509 certificate. This object should store the hash of the public X.509 certificate that the remote server should present during the (D)TLS connection setup. The fingerprint of the presented certificate and this hash value MUST match exactly, or the connection MUST NOT be established."

DEFVAL { "" }

::= { snmpTlstmAddrEntry 1 }

snmpTlstmAddrServerIdentity OBJECT-TYPE

SYNTAX SnmpAdminString

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The reference identity to check against the identity presented by the remote system."

DEFVAL { "" }

::= { snmpTlstmAddrEntry 2 }

snmpTlstmAddrStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The storage type for this conceptual row. Conceptual rows having the value 'permanent' need not allow write-access to any columnar objects in the row."

DEFVAL { nonVolatile }

::= { snmpTlstmAddrEntry 3 }

snmpTlstmAddrRowStatus OBJECT-TYPE

SYNTAX RowStatus

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The status of this conceptual row. This object may be used to create or remove rows from this table.

To create a row in this table, an administrator MUST set this object to either createAndGo(4) or createAndWait(5).

Until instances of all corresponding columns are

appropriately configured, the value of the corresponding instance of the snmpTlstmAddrRowStatus column is notReady(3).

In particular, a newly created row cannot be made active until the corresponding snmpTlstmAddrServerFingerprint column has been set.

Rows MUST NOT be active if the snmpTlstmAddrServerFingerprint column is blank and the snmpTlstmAddrServerIdentity is set to '\*' since this would insecurely accept any presented certificate.

The snmpTlstmAddrServerFingerprint object MUST NOT be modified while the value of this object is active(1).

An attempt to set these objects while the value of snmpTlstmAddrRowStatus is active(1) will result in an inconsistentValue error."

```
::= { snmpTlstmAddrEntry 4 }
```

```
-- *****  
-- snmpTlstmNotifications - Notifications Information  
-- *****
```

```
snmpTlstmServerCertificateUnknown NOTIFICATION-TYPE  
OBJECTS { snmpTlstmSessionUnknownServerCertificate }  
STATUS current  
DESCRIPTION
```

```
"Notification that the server certificate presented by an  
SNMP over (D)TLS server was invalid because no configured  
fingerprint or CA was acceptable to validate it. This may  
be because there was no entry in the snmpTlstmAddrTable or  
because no path could be found to known Certification  
Authority.
```

```
To avoid notification loops, this notification MUST NOT be  
sent to servers that themselves have triggered the  
notification."
```

```
::= { snmpTlstmNotifications 1 }
```

```
snmpTlstmServerInvalidCertificate NOTIFICATION-TYPE  
OBJECTS { snmpTlstmAddrServerFingerprint,  
          snmpTlstmSessionInvalidServerCertificates}  
STATUS current  
DESCRIPTION
```

```
"Notification that the server certificate presented by an  
SNMP over (D)TLS server could not be validated even if the  
fingerprint or expected validation path was known. That is,  
a cryptographic validation error occurred during certificate
```

validation processing.

To avoid notification loops, this notification MUST NOT be sent to servers that themselves have triggered the notification."

::= { snmpTlstmNotifications 2 }

```
-- *****  
-- snmpTlstmCompliances - Conformance Information  
-- *****
```

snmpTlstmCompliances OBJECT IDENTIFIER ::= { snmpTlstmConformance 1 }

snmpTlstmGroups OBJECT IDENTIFIER ::= { snmpTlstmConformance 2 }

```
-- *****  
-- Compliance statements  
-- *****
```

snmpTlstmCompliance MODULE-COMPLIANCE

STATUS current

DESCRIPTION

"The compliance statement for SNMP engines that support the  
SNMP-TLS-TM-MIB"

MODULE

MANDATORY-GROUPS { snmpTlstmStatsGroup,  
snmpTlstmIncomingGroup,  
snmpTlstmOutgoingGroup,  
snmpTlstmNotificationGroup }

::= { snmpTlstmCompliances 1 }

```
-- *****  
-- Units of conformance  
-- *****
```

snmpTlstmStatsGroup OBJECT-GROUP

OBJECTS {

snmpTlstmSessionOpens,  
snmpTlstmSessionClientCloses,  
snmpTlstmSessionOpenErrors,  
snmpTlstmSessionAccepts,  
snmpTlstmSessionServerCloses,  
snmpTlstmSessionNoSessions,  
snmpTlstmSessionInvalidClientCertificates,  
snmpTlstmSessionUnknownServerCertificate,  
snmpTlstmSessionInvalidServerCertificates,  
snmpTlstmSessionInvalidCaches

}

STATUS current

DESCRIPTION

```
    "A collection of objects for maintaining
    statistical information of an SNMP engine that
    implements the SNMP TLS Transport Model."
 ::= { snmpTlstmGroups 1 }
```

```
snmpTlstmIncomingGroup OBJECT-GROUP
```

```
  OBJECTS {
    snmpTlstmCertToTSNCount,
    snmpTlstmCertToTSNTableLastChanged,
    snmpTlstmCertToTSNFingerprint,
    snmpTlstmCertToTSNMapType,
    snmpTlstmCertToTSNData,
    snmpTlstmCertToTSNStorageType,
    snmpTlstmCertToTSNRowStatus
  }
  STATUS      current
  DESCRIPTION
    "A collection of objects for maintaining
    incoming connection certificate mappings to
    tmSecurityNames of an SNMP engine that implements the
    SNMP TLS Transport Model."
 ::= { snmpTlstmGroups 2 }
```

```
snmpTlstmOutgoingGroup OBJECT-GROUP
```

```
  OBJECTS {
    snmpTlstmParamsCount,
    snmpTlstmParamsTableLastChanged,
    snmpTlstmParamsClientFingerprint,
    snmpTlstmParamsStorageType,
    snmpTlstmParamsRowStatus,
    snmpTlstmAddrCount,
    snmpTlstmAddrTableLastChanged,
    snmpTlstmAddrServerFingerprint,
    snmpTlstmAddrServerIdentity,
    snmpTlstmAddrStorageType,
    snmpTlstmAddrRowStatus
  }
  STATUS      current
  DESCRIPTION
    "A collection of objects for maintaining
    outgoing connection certificates to use when opening
    connections as a result of SNMP-TARGET-MIB settings."
 ::= { snmpTlstmGroups 3 }
```

```
snmpTlstmNotificationGroup NOTIFICATION-GROUP
```

```
  NOTIFICATIONS {
    snmpTlstmServerCertificateUnknown,
    snmpTlstmServerInvalidCertificate
  }
```

```
STATUS current
DESCRIPTION
    "Notifications"
 ::= { snmpTlstmGroups 4 }
```

```
END
```

## 5. Security Considerations

This document updates a transport model that permits SNMP to utilize (D)TLS security services. The security threats and how the TLS transport model mitigates these threats are covered throughout this document and in [\[RFC6353\]](#). Security considerations for TLS are described in Section 10 and Appendix E of TLS 1.3 [\[RFC8446\]](#). Security considerations for DTLS are described in Section 11 of DTLS 1.3 [\[RFC9147\]](#).

Implementations should consider the latest recommendations on the use of (DTLS), such as that documented in [\[RFC9325\]](#).

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example, by using IPsec), there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is **RECOMMENDED** that only SNMPv3 messages using the Transport Security Model (TSM) or another secure-transport aware security model be sent over the TLSTM transport.

## 6. IANA Considerations

IANA is asked to create a new registry called the SNMP-TLSTM HashAlgorithm Registry in the Structure of Management Information (SMI) Numbers (MIB Module Registrations) Group and to update the proposed URL reference in the above MIB (listed as "https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml" under SnmpTLSTMFingerprint), if needed, to accurately reflect its location. The description of this registry should be:  
iso.org.dod.internet.mgmt.mib-2.snmpTlstmMIB.snmpTlstmHashAlgorithms (1.3.6.1.2.1.198.4)

The registry should have the following fields: value, description, recommended, and reference. The range of values is zero to 255, with initial assignments shown in Section 2.1. The "recommended" column indicates "Y" for hashing algorithms that are standards track and are deemed to be acceptable for widely applicable current use and "N" for hashing algorithms that reflect meanings that are not recommended (e.g., they do not provide sufficient security for modern systems, they are not standards track, they have limited applicability). A blank field indicates that no recommendation is made (e.g., because the value is reserved or left for private use).

This registry is expected to be updated infrequently and, as such, its values are limited to one octet.

The policy for updates is Expert Review. The expert should consult the Security Area, e.g. via the mailing list of the TLS WG (the initial values of this registry are taken from an existing TLS Registry so the TLS WG would seem the best fit for this).

While future additions to the IANA TLS HashAlgorithm Registry are not expected, any future addition to the IANA TLS HashAlgorithm Registry **MUST** be consistent with the values assigned in the IANA SNMP-TLSTM HashAlgorithm Registry.

## 7. Acknowledgements

This document is based on [RFC6353]. This document was reviewed by the following people who helped provide useful comments: Michaela Vanderveen, Joe Clarke, Jurgen Schonwalder, and Tom Petch.

## 8. References

### 8.1. Normative References

- [RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <<https://www.rfc-editor.org/info/rfc1123>>.
- [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>>.
- [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>>.
- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <<https://www.rfc-editor.org/info/rfc5890>>.
- [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>>.
- [RFC6353] Hardaker, W., "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", STD 78, RFC 6353, DOI 10.17487/RFC6353, July 2011, <<https://www.rfc-editor.org/info/rfc6353>>.

**[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>>.

**[RFC8446]**

Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <<https://www.rfc-editor.org/info/rfc8446>>.

**[STD58]**

McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIV2)", STD 58, RFC 2578, April 1999.  
McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIV2", STD 58, RFC 2579, April 1999.  
McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Conformance Statements for SMIV2", STD 58, RFC 2580, April 1999.

**[STD62]**

Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.  
Case, J., Harrington, D., Presuhn, R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3412, December 2002.  
Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, December 2002.  
Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.  
Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.  
Presuhn, R., Ed., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, December 2002.  
Presuhn, R., Ed., "Transport Mappings for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3417, December 2002.  
Presuhn, R., Ed., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

**8.2. Informative References**



- [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>>.
- [RFC5953] Hardaker, W., "Transport Layer Security (TLS) Transport Model for the Simple Network Management Protocol (SNMP)", RFC 5953, DOI 10.17487/RFC5953, August 2010, <<https://www.rfc-editor.org/info/rfc5953>>.
- [RFC8422] Nir, Y., Josefsson, S., and M. Pegourie-Gonnard, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier", RFC 8422, DOI 10.17487/RFC8422, August 2018, <<https://www.rfc-editor.org/info/rfc8422>>.
- [RFC8447] Salowey, J. and S. Turner, "IANA Registry Updates for TLS and DTLS", RFC 8447, DOI 10.17487/RFC8447, August 2018, <<https://www.rfc-editor.org/info/rfc8447>>.
- [RFC8996] Moriarty, K. and S. Farrell, "Deprecating TLS 1.0 and TLS 1.1", BCP 195, RFC 8996, DOI 10.17487/RFC8996, March 2021, <<https://www.rfc-editor.org/info/rfc8996>>.
- [RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <<https://www.rfc-editor.org/info/rfc9147>>.
- [RFC9150] Cam-Winget, N. and J. Visoky, "TLS 1.3 Authentication and Integrity-Only Cipher Suites", RFC 9150, DOI 10.17487/RFC9150, April 2022, <<https://www.rfc-editor.org/info/rfc9150>>.
- [RFC9325] Sheffer, Y., Saint-Andre, P., and T. Fossati, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 9325, DOI 10.17487/RFC9325, November 2022, <<https://www.rfc-editor.org/info/rfc9325>>.

#### Author's Address

Kenneth Vaughn (editor)  
Trevilon LLC  
1060 Highway 107 South  
Del Rio, TN 37727  
United States of America

Phone: [+1 571 331 5670](tel:+15713315670)

Email: [kvaughn@trevilon.com](mailto:kvaughn@trevilon.com)