IEEE P802.1AE/D?

Draft Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security

Sponsor

LAN MAN Standards Committee
of the
IEEE Computer Society

This individual contribution to the Project under consideration by the Link Security Task Group of IEEE 802.1 is intended to facilitate future progress. ***It has no official standing whatsoever and has not been reviewed by the task group.***

Abstract: This standard specifies how all or part of a bridged local or metropolitan area network can be secured transparently to communicating peers. MAC security (MACsec) entities in end stations and in MAC Bridges use secure associations and media access independent protocols to provide connectionless user data confidentiality, frame data integrity, and data origin authenticity between authorized ports.
Keywords: For keywords refer to the title page proper, following the editors' foreword.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEEE Standard is wholly voluntary. The IEEE disclaims liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEEE Standard document.

The IEEE does not warrant or represent the accuracy or content of the material contained herein, and expressly disclaims any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEEE Standards documents are supplied "**AS IS**."

The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Nor is the IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

IEEE is the sole entity that may authorize the use of certification marks, trademarks, or other designations to indicate compliance with the materials set forth herein.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (978) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Editors' Foreword

<<Notes>>

<<Throughout this document, all notes such as this one, presented between angle braces, are temporary notes inserted by the Editors for a variety of purposes; these notes and the Editors' Foreword will all be removed prior to publication and are not part of the normative text.>>

<<Comments and participation in 802.1 standards development

Comments on this draft are encouraged. **PLEASE NOTE: All issues related to IEEE standards presentation style, formatting, spelling, etc. are routinely handled between the 802.1 Editor and the IEEE Staff Editors prior to publication, after balloting and the process of achieving agreement on the technical content of the standard is complete.** Readers are urged to devote their valuable time and energy only to comments that materially affect either the technical content of the document or the clarity of that technical content. Comments should not simply state what is wrong, but also what might be done to fix the problem.

Full participation in the development of this draft requires individual attendance at IEEE 802 meetings. Information on 802.1 activities, working papers, and email distribution lists etc. can be found on the 802.1 website:

http://ieee802.org/1/

Use of the email distribution list is not presently restricted to 802.1 members, and the working group has had a policy of considering ballot comments from all who are interested and willing to contribute to the development of the draft. Individuals not attending meetings have helped to identify sources of misunderstanding and ambiguity in past projects. Non-members are advised that the email lists exist primarily to allow the members of the working group to develop standards, and are not a general forum.

Comments on this document may be sent to the 802.1 email exploder, to the editors, or to the Chairs of the 802.1 Working Group and Link Security Task Group. :

Dolors Sala
Chair, 802.1 Link Security Task Group

Email:dolors@ieee.org

Tony Jeffree
Chair, 802.1 Working Group
11A Poplar Grove
Sale
Cheshire
M33 3AX
UK
+44 161 973 4278 (Tel)
+44 161 973 6534 (Fax)
Email: tony@jeffree.co.uk

PLEASE NOTE: Comments whose distribution is restricted in any way cannot be considered, and may not be acknowledged.
>>

<<The draft text and accompanying information

This document currently comprises:

- A temporary cover page, preceding the Editors' Forewords. This cover page will be removed following working group approval of this draft, i.e. prior to sponsor ballot.
- IEEE boilerplate text.
- The editors' forewords, including this text. These include an unofficial and informal appraisal of history and status, introductory notes to each draft that summarize the progress and focus of each successive draft, and requests for comments and contributions on major issues.
- A title page for the proposed standard including an Abstract and Keywords. This title page will be retained following approval.
- IEEE boilerplate text (identical to the above).
- An introduction to the family of 802 standards.
- The introduction to this standard, as revised by this proposed draft. This follows the preceding item and is actually important.
- A record of participants (not included in early drafts but added prior to publication).
- The proposed revision proper.
- An Annex Z comprising the editors' discussion of issues. This annex will be deleted from the document prior to sponsor ballot.

During the early stages of draft development, 802.1 editors have a responsibility to attempt to craft technically coherent drafts from the resolutions of ballot comments and the other discussions that take place in the working group meetings. Preparation of drafts often exposes inconsistencies in editors instructions or exposes the need to make choices between approaches that were not fully apparent in the meeting. Choices and requests by the editors' for contributions on specific issues will be found in the editors' introductory notes to the current draft, at appropriate points in the draft, and in Annex Z. Significant discussion of more difficult topics will be found in the last of these.

The ballot comments received on each draft, and the editors' proposed and final disposition of comments, are part of the audit trail of the development of the standard and are available, along with all the revisions of the draft on the 802.1 website (for address see above).

>>

<<History and Scope

A PAR for this project was drafted at the June 2003 802.1 interim meeting, and has been submitted to the 802 SEC for circulation to and to solicit comment from other P802 Working Groups. It is anticipated that a final proposed PAR will be forwarded for SEC consideration by vote of the 802.1 Working Group at its closing plenary during the July 2003 meeting of P802.

>>

<<Introductory notes to the current draft

This document, P802.1AE/D?, is not a working group or task group draft, but an individual contribution intended to facilitate future progress.
>>

<<Notes to prior drafts (excerpts of continuing relevance).

P802.1D/D0:

P802.1/D0 was the first of the full revision series. The prior development of P802.1y is archived on the IEEE 802.1 website, and the notes in Annex Z summarize prior technical progress.

In July 2002 we discussed what we might do if there were objections to removing the STP material in clause 8, on the grounds that it should still be accessible for historical reasons. Having worked on the draft it seems clear that not removing the material would either be an obstacle to correctly expressing the technical content of newer material or necessitate a thorough revision of the way the old material fits into the document. It seems unlikely that we would expend the effort or have the enthusiasm to do the latter well. Moreover cutting out dead wood references to withdrawn standards creates its own problem of historical reference, while leaving it in carries forward a maintenance load. The only sensible approach to this would seem to be to retain an archive of withdrawn and superseded copies of our standards, accessible to those who have a need to retrieve historical information. In turn this helps us cut out further dead wood, saving the reader from plowing through irrelevancies, and helping maintenance.
>>

<<Editors' final checklist (items noted in development, to be applied to final text.

The published standards are inconsistent and a bit of a mess when it comes to PDF bookmarks, this makes using them rather than final working group text difficult. P802.1p/D9 was very good. In particular it provides bookmarks for all figures at the end of a clause (see clause 7 for an example), need to copy that example.
>>

IEEE P802.1AE/D?

Draft Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security

Sponsor

LAN MAN Standards Committee
of the
IEEE Computer Society

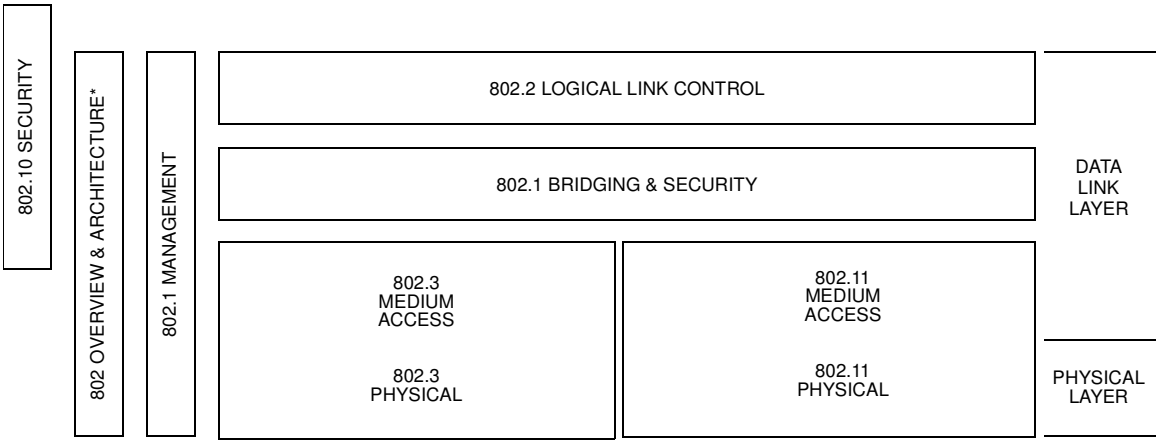
Contribution to the Link Security Task Group of IEEE 802.1

Abstract: This standard specifies how all or part of a bridged local or metropolitan area network can be secured transparently to communicating peers. Secure associations, between protocol entities in end stations and in MAC Bridges, are used by media access independent protocols to support connectionless user data confidentiality, frame data integrity, and data origin authenticity between authorized ports.
Keywords: local area networks, LANs, metropolitan area networks, MANs, security, MAC security confidentiality, integrity, data origin authenticity, port based network access control, MAC Service, MSAP, service access point, transparent bridging, MAC Bridges, port based network access control, authorized port, secure association.

Introduction to IEEE 802 Local And Metropolitan Area Network Standards

[This introduction is not part of IEEE Std 802.1AE, 200X Edition, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security.]

This standard is part of a family of standards for local and metropolitan area networks. The relationship between the standard and other members of the family is shown below. (The numbers in the figure refer to IEEE standard numbers.)



The IEEE 802 family of standards deals with the Physical and Data Link layers as defined by the International Organization for Standardization (ISO) Open Systems Interconnection (OSI) Basic Reference Model (ISO/IEC 7498-1 : 1994). The access standards define medium access technologies and associated physical media, each appropriate for particular applications or system objectives.

The standards defining the technologies noted above are as follows:

- IEEE Std 802 *Overview and Architecture.* This standard provides an overview to the family of IEEE 802 Standards.
- IEEE Std 802.1B *LAN/MAN Management.* Defines an OSI management-compatible architecture, and services and protocol elements for use in a LAN/MAN environment for performing remote management.
- IEEE Std 802.1D *Media Access Control (MAC) Bridges.* Specifies an architecture and protocol for the interconnection of IEEE 802 LANs below the MAC service boundary.
- IEEE Std 802.1AE *Media Access Control (MAC) Security.* Specifies an architecture and protocol for IEEE 802 LAN Security.
- IEEE Std 802.1F Common Definitions and Procedures for IEEE 802 Management Information
- IEEE Std 802.2 Logical link control
- IEEE Std 802.3 CSMA/CD access method and physical layer specifications
- IEEE Std 802.11 Wireless LAN Medium Access Control (MAC) and physical layer specifications

IEEE Std 802.AE, 200X Edition

.....

Relationship between IEEE Std 802.1AE and other IEEE Std 802.1 standards

IEEE Std 802.1X specifies...

IEEE Std 802.1A? specifies key management and the establishment of secure associations used by IEEE Std 802.1AE.

Contents

IEEE P802.1AE/D?

Draft Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security

1. Overview

1.1 Introduction

IEEE 802 Local Area Networks (LANs) are often used...

<<in applications where security matters and there is a security threat... an informal and easy to read description in one paragraph of no more than 5 lines>>

MAC Security, as defined by this Standard, allows...

<< an informal and easy to relate introduction to the effects and benefits of MAC Security in one paragraph of not more than 5 lines>>

MAC Security provides for

<<between 3 and a maximum of 8 specific bullet points to backup the preceding paragraph>>

a) specific benefit 1
b) specific benefit 2
c) specific benefit 3

1.2 Scope

The scope of the standard, as stated in the PAR, is

> To specify provision of connectionless user data confidentiality, frame data integrity, and data origin authenticity by media access independent protocols and entities that operate transparently to MAC Clients*. Key management and the establishment of secure associations is outside the scope but will be referenced by this project.
>
> *As specified in IEEE Standards 802, 802.2, 802.1D, 802.1Q, and 802.1X.

This standard specifies provision of connectionless user data confidentiality, frame data integrity, and data origin authenticity by media access independent protocols and entities that operate transparently to MAC Clients. To this end it

a) Specifies the requirements for MAC Security in terms of provision of the MAC Service and the preservation of the semantics and parameters of service requests and indications.
b) Describes the threats, both intentional and accidental, to correct provision of the service.
c) Specifies the security services supported by MAC Security to prevent, mitigate, or restrict the impact or scope of attacks that present these threats.

d) Examines the potential impact of both the threats and the use of MAC Security on the Quality of Service, specifying constraints on the design and operation of entities and protocols that provide MAC Security.
e) Models support of the secure MAC Service in terms of the operation of media access control method independent MAC Security Entities (SecYs) within the MAC Sublayer in End Stations and Bridges.
f) Identifies the functions to be performed by each SecY, and provides an architectural model of its internal operation in terms of Processes and Entities that provide those functions.
g) Establishes the requirements for security wrapper to provide security services for each MAC Protocol Data Unit (MPDU) delivered or accepted by a SecY to or from client entities and communicated to or from peer SecYs.
h) Establishes the requirements for a protocol between peer SecYs to identify the potential end points of the secure associations (SAs) used by security wrappers <<improve wording>>.
i) Specifies the interface/exchanges between a secY and its associated and collocated Port Access Entity (PAE) that provides and updates cryptographic keying and secure association identification (SAID) information for the SecY.
j) Establishes the requirements for management of MAC Security, identifying the managed objects and defining the management operations for MAC Security Entities.
k) Specifies performance requirements and recommends default values and applicable ranges for the operational parameters of a SecY.
l) Specifies the requirements to be satisfied by equipment claiming conformance to this standard.

<<what else?>>

This standard does not

m) specify key management or key distribution protocols, but makes use of...

<<find words for this>>

n) blah...

<<what else?>>

2. References

The following standards contain provisions which, through reference in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of ISO and IEC maintain registers of currently valid International Standards.

ANSI X3.159-1989, American National Standards for Information Systems—Programming Language—C.[1]

IEEE Std 802-2001, IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture.[2]

IEEE Std 802.1D-2003, IEEE Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges.

IEEE Std 802.1Q-2003, IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks.

IEEE Std 802.1X-2001, IEEE Standards for Local and Metropolitan Area Networks—Port Based Network Access Control.

IEEE Std 802.2, 1998 Edition [ISO/IEC 8802-2: 1998], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 2: Logical link control.[3]

IEEE Std 802.3, 2002 Edition, IEEE Standards for Local and Metropolitan Area Networks, Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications: Aggregation of Multiple Link Segments.

IEEE Std 802.11, 1999 Edition [ISO/IEC 8802-11: 1999], Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.

<<802.1aa if not yet a standard, will be removed before final draft>>

IETF RFC XXXX[4]

ISO 6937-2: 1983, Information processing—Coded character sets for text communication—Part 2: Latin alphabetic and non-alphabetic graphic characters.[5]

---

[1]ANSI publications are available from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

[2]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA. IEEE publications can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org

[3]ISO [IEEE] and ISO/IEC [IEEE] documents are available from ISO Central Secretariat, 1 rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse; and from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA. ISO [IEEE] and ISO/IEC [IEEE] documents can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org.

[4]Internet RFCs are available from the Internet Engineering Task Force website at http://www.ietf.org/rfc.html.

[5]ISO and ISO/IEC documents are available from the ISO Central Secretariat, 1 rue de Varembé, Case Postale 56, CH-1211, Genève 20, Switzerland/Suisse; and from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA.

ISO/IEC 7498-1: 1994, Information processing systems—Open Systems Interconnection—Basic Reference Model—Part 1: The Basic Model.

ISO/IEC TR 11802-2: 1997, Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Technical reports and guidelines—Part 2: Standard Group MAC addresses.

ISO/IEC 14882: 1998, Information Technology—Programming languages—C++.

ISO/IEC 15802-1: 1995, Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Common specifications—Part 1: Medium Access Control (MAC) service definition.

3. Definitions

For the purposes of this standard, the following terms and definitions apply.

<<This is for LinkSec - both MACsec and Keysec- see RFC 2401 (limited usefulness), IPsec; RFC 2828 (useful); 802.10 Definitions and Acronyms; 802.11i Definitions and Acronyms; Moskowitz, Authentication Types Memo>>

3.1 authentication: The process of verifying an identity claimed by or for a system entity. See data origin authentication and peer entity authentication. Includes both message authentication and data origin authentication.

An authentication process consists of two steps:

1) Identification step: Presenting an identifier to the security system. (Identifiers should be assigned carefully, because authenticated identities are the basis for other security services, such as access control service.)
2) Verification step: Presenting or generating authentication information that corroborates the binding between the entity and the identifier. (RFC 2828)

3.1.1 message authentication: If the message arrives authenticated, the cryptographic guarantee is that the message has not been tampered with.

3.1.2 data origin authentication: The corroboration that the source of the data recieved is as claimed. The message really did originate with the purported sender, in the cryptographic sense that the sender possesses the key. In the case of symmetric keys, the origin is defined as the entity that possesses the key. In the point to point case, only two parties have the key. If the message arrives authenticated, the receiver knows that it must come either from himself or from the purported sender. Data origin authentication does not include non-repudiation, as does digital signature, which is a stronger form of authentication of the sender.

3.1.3 asymmetric authentication - Using an underlying mathematical concept that is asymmetric. The key that is used to decrypt is different than key use to encrypt the message, and it is mathematically awkward to derive one from the other. Public key encryption, for example Diffie-Hellman or RSA.

3.1.4 symmetric authentication: One key does everything, both encrypt and decrypt. For example, challenge response methods, such as that used in One Time Password OTP are based on symmetric authentication. Given a random challenge, A transforms data with A's key and a crypto function. C performs the same transform with C's key and confirms that the result is the same. Can use any crypto function, for example DES, SHA1.

3.1.5 peer-entity authentication: The corroboration that a peer entity in an association is the one claimed. This service, when provided by the (N)-layer, provides corroboration to the (N+1)-entity that the peer entity is the claimed (N+1)-entity. (ISO/IEC 7498-2: 1989) Note: This is primarily intended for, although not limited to, connection-oriented service and may be either unilateral or mutual. (.10)

3.2 birthday attack: A class of brute-force techniques used in an attempt to solve a class of cryptographic hash function problems. These methods take advantage of functions which, when supplied with a random input, return one of k equally likely values. By repeatedly evaluating the function for different inputs, the same output is expected to be obtained after about 1.2 sqrt(k) evaluations. (mathworld.wolfram.com)

3.3 bridged local area network: A concatenation of individual IEEE 802 LANs interconnected by MAC Bridges.

NOTE—Unless explicitly specified the use of the word 'network' in this Standard refers to a Bridged Local Area Network. The term Bridged Local Area Network is not otherwise abbreviated. The term Local Area Network and the abbreviation LAN are used exclusively to refer to an individual LAN specified by a MAC technology without the inclusion of Bridges. This precise use of terminology within this specification allows a Bridged Local Area Network to be distinguished from an individual LAN that has been bridged to other LANs in the network. In more general usage such precise terminology is not required, as it is a explicit goal of this standard that MAC Security is transparent to the users of the MAC Service.

3.4 cipher suite: A set of one or more algorithms, designed to provide data confidentiality, data authenticity or integrity, and/or replay protection. (.11)

3.5 client: A client of an N-entity is the N-Service User of the service provided by that entity.

3.6 controlled port: <<See Section 6.4 in 802.1aa/D6>> Port Access Control has the effect of creating two distinct points of access to the Authenticator System's point of attachment to the LAN. One point of access allows the uncontrolled exchange of PDUs between the System and other Systems on the LAN, regardless of the authorization state (the uncontrolled Port); the other point of access allows the exchange of PDUs only if the current state of the Port is Authorized (the controlled Port). The uncontrolled and controlled Ports are considered to be part of the same point of attachment to the LAN; any frame received on the physical Port is made available at both the controlled and uncontrolled Ports, subject to the authorization state associated with the controlled Port.

3.7 uncontrolled port:

3.8 data integrity: The condition or state in which data has not been altered or destroyed in an unauthorized manner. (ISO/IEC 7498-2: 1989) (.10)>>

3.9 connectionless data confidentiality: The protection of (N)-service data units from unauthorized disclosure during transmission from one (N+1)-entity to one or more (N+1)-entities, where there is no reliance of one protected PDU on any of its predecessors.

3.9.1 connectionless integrity: A service providing for the integrity of a single MSDU, without reliance on predecessors. It may take the form of determining whether or not the received MSDU has been modified. (.10)

3.9.2 connection-oriented confidentiality:

3.9.3 connection-oriented integrity:

3.10 digital signature: Message integrity function using public key algorithms. The guarantee is that the message absolutely originated with the sender, unless he gave away his private key. Includes non-repudiation.

3.11 identity:

3.12 initialization vector (IV): A random binary sequence used at the beginning of a cryptographic operation to allow cryptographic chaining.

3.13 integrity check value (ICV): A value that is derived by performing an algorithmic transformation on the data unit for which data integrity services are provided. The ICV is sent with the protected data unit and is recalculated and compared by the receiver to detect data modification. MAC Message Authentication Code and MIC Message Integrity Code are synonyms

3.14 key: A sequence of octects that controls the operation of cryptographic functions.

3.14.1 pre-shared key: A static key that is distributed to the units in the system by out-of-band means. (.11)

3.14.2 secret key: The traditional cryptographic key known only to the communicating parties and used for both encipherment and decipherment. (.10)

3.14.3 key management: The generation, storage, distribution, deletion, archiving, and application of keys in accordance with a security policy. (ISO/IEC 7498-2: 1989) (.10)

3.15 IEEE 802 Local Area Network (LAN): IEEE 802 LANs (also referred to in the text simply as LANs) are LAN technologies that provide a MAC Service equivalent to the MAC Service defined in ISO/IEC 15802-1. IEEE 802 LANs include IEEE Std 802.3 (CSMA/CD), and IEEE Std 8802.11 (Wireless).

3.16 MAC Media Access Control:

3.17 MACsecY:

3.18 Message Integrity Code (MIC): A value generated by a symmetric cryptographic function. If the input data is changed, a value cannot be correctly computed without knowledge of the secret key. Thus, the secret key protects the input data from undetectable alteration. This is traditionally called a Message Authentication Code, or MAC, but the acronym MAC is already reserved for another meaning in this standard (.11) See ICV.

3.19 MAC service data unit (MSDU): MAC data originating from the user of the protocol.

3.20 mode: A mode of operation, or mode, for short, is an algorithm that features the use of a symmetric key block cipher algorithm to provide an information service, such as confidentiality or authentication. (CSRC)

3.21 nonce: A value that shall not be reused with a given key, including over all re-initializations of the system through all time. If the key lifetime is long term, in particular, across system reboots, then any nonces must be unique and not used across system reboots. Means “the value of n once”, this number n is used only once ever. Usually appears at the beginning of a packet. If the nonce were re-used, it would break the security of the cryptographic algorithm.

3.22 non-repudiation:

3.23 port access entity (PAE): The protocol entity associated with a Port. It can support the protocol functionality associated with the Supplicant the Authenticator, or both.

3.24 protocol data unit (PDU): A unit of data specified in a protocol and consisting of protocol information and, possibly, user data. (ISO/IEC 7498-1: 1994) (.10)

3.25 security association (SA): A cooperative relationship between entities formed by the sharing of cryptographic keying information and security management objects.

3.26 hierarchy of security associations:

3.27 security association identifier (SAID): A specific security association is usually identified by carrying a Security Association Identifier (SAID) in a special field. Conceptually, this SAID is an index into a database maintained at either end of a Security Association that contains the relevant security parameters. .

3.28 security tag:

3.29 sequence number: A monotonically increasing value used to uniquely identify frame in sequence of frames. Never repeated in the lifetime of a key. The sequence number could be used as a nonce by some algorithms that do not require the value to also be unpredictable.

3.30 spoofing: Claiming a fraudulent identity for purposes of mounting an attack

3.31 trust: When one party trusts the other party to not subvert the goals of the protocols, e.g., it will not attempt to perform the following attacks: spoofing, repudiation, information disclosure, denial of service, or elevation of privilege.

4. Abbreviations

The following abbreviations are used in this standard.

AAD	Additional Authenticated Data
AES	Advanced Encryption Standard
AS	Authentication Server
CBC	Cipher-Block Chaining
CBC-MAC	CBC Message Authentication Code.
CCM	Counter mode with CBC-MAC
CCMP	CCM Protocol
CRC	Cyclic Redundancy Check
CTR	Counter mode
EA	Crypto algorithm that does both encryption and authentication
FCS	Frame Check Sequence
FIPS	Federal Information Processing Standard
ICV	Integrity Check Value
kb/s	Kilobit per second (1 kb/s is equivalent to 1000 bits per second)
MAC	Media Access Control
MIC	Message Integrity Code
Mb/s	Megabit per second (1 Mb/s is equivalent to 1,000,000 bits per second)
MSDU	MAC Service Data Unit
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OUI	Organizationally Unique Identifier
PAE	Port Access Entity
PBL	Packet Body Length (IEEE 802.1X)
PDU	Protocol Data Unit
PN	Packet Number

PMK	Pairwise Master Key
PRF	Pseudo-Random Function
PRNG	Pseudo Random Number Generator
PSK	Pre-Shared Key
RSN	Robust Security Network
RSTP	Rapid Spanning Tree Algorithm and Protocol
RST BPDU	Rapid Spanning Tree Bridge Protocol Data Unit
SNAP	Sub-Network Access Protocol
SAID	Secure Association IDentifier
[SPI	Security parameters index from IPsec]
TLS	Transport Layer Security
Tb/s	Terabit per second (1 Tb/s is equivalent to 1,000,000 MB/s)

5. Conformance

5.1 Required Capabilities

An implementation of a MAC Security Entity (MACsecY) for which conformance to this standard is claimed shall

a) Implement the MAC Security Protocol (MACsec), as specified in Clause 9.
b) Encode transmitted SMPDUs and validate received SMPDUs as specified in Clause 9.
c) Specify the following parameters of the implementation
    1) Filtering Database Size, the maximum number of entries.
    2) Permanent Database Size, the maximum number of entries.
d) Specify the following performance characteristics of the implementation
    1) .....
    2) .....

5.2 Optional Capabilities

An implementation of a MAC Security Entity (MACsecY) for which conformance to this standard is claimed may

a) .....
b) .....

NOTE—The term capability is used to describe a set of related detailed provisions of this Standard. Each capability can comprise both mandatory provisions, required if implementation of the capability is to be claimed, and optional provisions. Each detailed provision is specified in on or more of the other clauses of this standard. The PICS, described below, provides a useful checklist of these provisions.

5.3 Protocol Implementation Conformance Statement

The supplier of an implementation that is claimed to conform to this standard shall complete a copy of the PICS proforma provided in Annex A (normative) and shall provide the information necessary to identify both the supplier and the implementation.

6. Secure support of the MAC Service

MACsec provides security for communications between authenticated and authorized peer MAC Security Entities attached to and providing service at service access points in

a) End stations attached to Local Area Networks; and
b) MAC Bridges that connect Local Area Networks into a Bridged Local Area Network.

This clause discusses the following aspects of secure service provision

a) Provision of the MAC Service to MAC Service users in End Stations
b) Provision of the MAC Internal Sublayer Service to MAC Relay Entities in MAC Bridges
c) Provision of the MAC Enhanced Internal Sublayer Service in VLAN-aware Bridges
d) Preservation of the MAC Service
e) Security threats
f) Maintenance of Quality of Service.

This clause discusses threats against the MAC Service, and it also considers how providing security to mitigate these attacks may itself affect the MAC Service.

<<What are the defining characteristics of the MAC Service? To what threats are these services vulnerable? What security features can be provided to prevent, mitigate, these attacks? How can these security services affect the MAC Service?>>

6.1 Provision of the MAC Service

The service provided to End Stations is the (unconfirmed) connectionless-mode MAC Service defined in ISO/IEC 15802-1. The MAC Service is defined as an abstraction of the features common to a number of specific MAC Services; it describes the transfer of user data between source and destination end stations, via MA-UNITDATA request primitives and corresponding MA-UNITDATA indication primitives issued at MAC service access points. Each MA-UNITDATA request and indication primitive has four parameters: Destination Address, Source Address, MAC Service data unit (MSDU), and Priority.

The service provided to the MAC Relay Entity of a MAC Bridge is the MAC Internal Sublayer Service (ISS) specified in Clause 6.4 of IEEE Std 802.1D. The ISS is derived from the MAC Service by augmenting the specification of the MA_UNITDATA.request and MA_UNITDATA.indication primitives with two parameters necessary to the performance of the relay function. These are frame_type and frame_check_sequence. The definition of the ISS does not add any new service primitives to those defined by the LAN MAC Service Definition.

The service provided to the MAC Relay Entity of a VLAN-aware MAC Bridge is the MAC Enhanced Internal Sublayer Service (EISS) specified in Clause 6.4 of IEEE Std 802.1Q. The EISS is derived from the ISS by augmenting the specification of the M_UNITDATA.request and M_UNITDATA.indication primitives with parameters necessary to the operation of the tagging and untagging functions of the VLAN Bridge. A canonical_format_indicator, vlan_classification, option rif_information, and an include_tag are added to the request primitive. A canonical_format_indicator, vlan_identifier, and optional rif_information are added to the indication primitive. The definition of the EISS does not add any new service primitives to those defined by the ISS.

End stations and MAC Bridges may restrict the provision of the service, i.e., may restrict the service access points at which request primitives can be issued and corresponding indication primitives can occur, to those that are bound to authenticated and authorized devices. Unauthorized devices can be denied access to a LAN

or to a Bridged Local Area Network, other than to allow protocol exchanges required by an authentication process.

<<'bound to a device' means accessible only by entities within that device.>>

<<The above specifications are not complete for the purposes of security - some fundamental properties of the service (such as symmetrical and transitive communication) need to be added as described below.>>

<<There is an approved PAR for (improving) the specification of the MAC Service (P802.1ac).>>

<<Should mention MAC status parameters (802.1D Clauses 6.4.2, 6.4.3) somewhere, are they a part of the service that needs to be explicitly provided, preserved, and supported from the security point of view? Perhaps MAC Operational needs to be set to reflect temporary failures of the secure support to preserve the necessary connectivity.>>

6.1.1 Point to Point Service

Bridged LAN provides for a point to point topology. This topology offers secure communication naturally in the sense that eavesdropping is not possible except under certain types of attacks, as discussed below. The principal (only??) attack on a point to point topology is wiretapping, which is addressed by LinkSec.

<<security properties of the topology>>

<<The MAC service for wired is defined in 802.1D, constrained to that definition. A bridge does not forward a frame other than to the destination MAC address. Therefore, the shared media ability to read the frame anywhere is contravened. The only service a higher layer can depend on is that defined in802.1D, which is delivery to the specified MAC destination address.>>

6.1.2 Point to Multi-point Service

<<where does multi-hop fit in?>>

<<Multi-point service needs to be defined, probably in P802.1ac, and discussed.>>

EPON provides point to multi-point topology. This communication model is inherently insecure, as eavesdropping in the downstream direction is possible. Messages sent to a particular station can be read by another station to which the message was not sent

<<In an EPON the ability of a station to read a message not addressed to it, is a security threat. The countermeasure is to make the message unintelligible except to the destination addressee - i.e., encryption.>>

6.1.3 Multi-point to Multi-point, Shared Media Service

Multi-point to multi-point topology is the same communication model as shared media. Provider Bridges using VLANs provide a shared media topology. This communication model is inherently insecure as eavesdropping between any two stations is possible. Shared media, yellow coax (or wireless), messages can be read everywhere.

<<can specify the threats as put forth by 802.10>>

NOTE—Authentication and authorization is outside the scope of this standard, which just ensures secure communication between mutually authenticated and authorized service access points. (??)

6.2 Preservation of the MAC service

Preservation of the MAC Service comprises both

a) delivering the ‘correct’ parameters on a service indication, and
b) causing service indications to occur at the ‘correct’ destinations in the network.

The MAC Service consists of important and expected features. Attacks, or security breaches, prevent or distort the delivery of these features. Attacks may be intentional, perpetuated by an attacker, or they may come about accidentally through unintentional means, such as misconfiguration. Well-known security threats constitute attacks against provision of the expected MAC Service.

<<Specify important, interesting features of the MAC Service. What well-known attacks are threats to these services? How do threats jeopardize this service? What security countermeasures could prevent or tamper with each feature? >>

In providing safeguards or countermeasures, the MAC Service must be preserved. Security features must not alter or hamper the expected MAC Service, either in their design or implementation.

Providing security changes the expected delivery service, from a high probability of delivering “correct” frames to high probability of delivering “correct” and “secure” frames. A frame may be correct, e.g., it passes the CRC test, and yet it may not pass security checks, e.g., it could fail the ICV, or not have a recognizable SAID. If the frame fails security checks, it is not delivered.

It is also possible to deliver meaningless data. If the ICV is computed over encrypted data, then the frame may pass the ICV but the decrypted data could be meaningless. For example, the encrypted data might have been encrypted under a different key than the one the receiver expected.

Providing security may introduce random delay variation in transit times which would impact performance. The MAC Service would not be violated, but the performance could be affected.

The expected size of the frame will be changed, this is discussed further in cl 9 and clZ.

Race conditions in establishing secure connection must be avoided, or they can disrupt traffic.

<<How do the security mechanisms need to be designed in order to avoid breaking these services?>>

6.2.1 Deliver ‘correct’ parameters on a service indication

Preserving the MAC Service requires delivery of the ‘correct’ parameters on a service indication, i.e., the same values of the parameters as supplied by the corresponding service request. The parameters cannot undergo an unauthorized change. There may be specified exceptions in which specific parameters can be modified (or delivered with a certain probability of accidental modification) by the service provider. Each MA-UNITDATA request and indication primitive has four parameters: Destination Address, Source Address, MAC Service Data Unit (MSDU), and Priority. Each of these is further discussed as a sub clause.

<<Needs blow by blow, parameter by parameter discussion, some here and some under the QoS Maintenance heading. >>

6.2.2 Destination Address

An endpoint is uniquely identified. User data is delivered ONLY to specified endpoints. If it arrives elsewhere, there is a security breach.

A frame can arrive at an unintended destination through wiretapping, a man-in-the-middle attack, address spoofing the destination address.

In order to protect against such threats, the destination address must be integrity protected by the ICV. The destination address may be copied into an AAD field if necessary, depending on the cipher suite.

6.2.3 Source Address

Each endpoint is uniquely identified. User data comes ONLY FROM the place where the request is issued.

An attacker can take on the identity of an endpoint by means of a man-in-the-middle attack, masquerading, spoofing the source address, wiretapping.

Cryptographic techniques can protect against such threats. The source address must be integrity protected and may be copied into an AAD field if necessary, depending on the cipher suite.

6.2.4 MAC Service Data Unit (MSDU)

The MSDU should not undergo an unauthorized change or disclosure. .

An attacker can change an MSDU via a man-in-the-middle attack, wiretapping. Similarly, disclosure can be caused by the frame being delivered to an unintended recipient.

Unauthorized changes to the MSDU can be prevented with encryption and authentication. The MSDU field must be integrity protected and may be encrypted. Encryption will be negotiated as part of the Security Association. Disclosure needs to be prevented by encryption at the higher level, for example, with IPsec.

6.2.5 Priority

The Priority parameter should not undergo an unauthorized change. A network device may need to change the priority parameter in order to accomplish drop precedence.

<<Must be integrity protected, may be encrypted...?>>

6.2.6 Cause service indications to occur at the ‘correct’ places

Service indications need to occur at the ‘correct’ places. ‘Correct’ means, first, as constrained by the basic service specification - service access points compose a symmetric transitive group. Secondly, ‘correct’ means as required by security - only authenticated and authorized members are permitted in the group. Correct places can (by design or accident) be a subset of the places to which unsecured service can deliver.

Delivery must be to the intended destination address or addresses and not to other destinations.

Man-in-the-middle attacks can cause delivery to unintended destinations.

If MAC address tables become full, frames are sent to all stations. In this way a point to point network can be converted into a shared media network. Flooding a LAN with random MAC addresses will cause MAC address tables to overflow.

6.2.7 MAC Service is transitive with respect to connectivity

If station A can contact Station B, and if Station B can contact Station C, then Station A can contact Station C. This feature provides shared connectivity. Many network services depend on this feature, e.g., OSPF.<<more>>

Security Association may break transitivity. Authentication may break transitivity.

Some protocol features depend on the assumption of shared medium. An SA is point to point, unless we have group SAs, which is much more difficult.

<<How do we want to handle this? Detail the different cases where this issue arises.>>

6.2.8 MAC Service is symmetric with respect to connectivity

If station A can contact station B, then station B can contact station A.

<<The MAC Service provided by a Bridged Local Area Network is similar to that provided by a single LAN (6.3). In consequence

A Bridge is not directly addressed by communicating end stations, except as an end station for management purposes: frames transmitted between end stations carry the MAC Address of the peer-end station in their Destination Address field, not a MAC Address of the Bridge.

All MAC Addresses need to be unique within the network.

MAC Addresses of end stations are not restricted by the topology and configuration of the network.>>

Security Associations are uni-directional. In order to provide a two-way security association, two SAs must be established, from A==>B and from B==>A.

<<IPsec has a commit protocol, we need some functionally similar mechanism that will guarantee the maintenance of this property. Reboot situations cause temporary loss of synchrony between the stations. Key protocol should take care of this. If you receive traffic under an SA that you don't identify, this should trigger SA creation protocol. Traffic will be lost until a new SA is set up.>>

<<What are the implications of SAs for symmetric connectivity?>>

6.3 Quality of service maintenance

.Attacks and security breaches affect the quality of the MAC Service provided.

In addition, providing security must also make sure to preserve the quality of service.

Quality of Service comprises

a) Service availability
b) Frame loss
c) Frame mis-ordering
d) Frame duplication
e) Frame transit delay
f) Frame lifetime
g) Undetected frame error rate
h) Maximum service data unit size supported
i) Frame priority
j) Throughput

6.3.1 Service availability

Service availability is measured as a fraction of some total time during which the MAC Service is provided.

MACsec threats: The operation of MACsec has the potential to lower the service availability either from the normal operation of the security function, or through additional opportunities for illegitimate resource consumption. In the normal operation of MACsec, there will be a transition period when the MAC Service is not yet available because security association has not been established. This will reduce the amount of time that service is available, though by a small amount.

Because key agreement protocols are computationally expensive, and stations have limited computational resources, key management protocols present opportunities for additional vulnerabilities from attacks designed to consume resources. Key agreement protocols are resilient to some but not all of these forms of attack.

Well-known security threats: Service availability can be lowered by DoS attacks, which may be caused by masquerading, unauthorized data modification. Any time resources are used up, frames will eventually be dropped, causing a lowering of service availability.

6.3.2 Frame loss

The MAC Service does not guarantee the delivery of Service Data Units. Frames transmitted by a source station arrive, uncorrupted, at the destination station with high probability.

MACsec threats: The operation of MACsec introduces minimal additional frame loss. During the transition period while a security association is being set up, frames will have to either be held or dropped. Since buffer size is limited, they may be dropped.

Well-known security threats: A frame transmitted by a source station can fail to reach its destination station as a result of two types of interference. DoS attacks can cause insufficient resources to be available to process frames, so they are dropped. Alternatively, if there are any changes to the frame that are discovered through the ICV or the CRC, then the frame is discarded at the receiver, thus also increasing the loss rate. Intereference from any source can generate frame loss.

6.3.3 Frame miss-ordering

The MAC Service (9.2 of ISO/IEC 15802-1) permits a negligible rate of reordering of frames with a given user priority for a given combination of destination address and source address. MA_UNITDATA.indication service primitives corresponding to MA_UNITDATA.request primitives, with the same requested priority and for the same combination of destination and source addresses, are received in the same order as the request primitives were processed.

MACsec threats: There are no MACsec induced threats.

Well-known security threats: A man-in-the-middle could shuffle the frames and re-emit them in a different order. Because MAC sec provides frame by frame protection, it cannot protect the sequence of frames. it can only protect individual frames from threats.

6.3.4 Frame duplication

The MAC Service (9.2 of ISO/IEC 15802-1) permits a negligible rate of duplication of frames. MACsec protects against duplication by replay protection. If the higher layer protocol is not idempotent, then replay protection is needed at the L2 layer.

Sequence numbers in the frame that are integrity protected by the ICV provide upper and lower replay windows. See appendix Z. If an attacker changes the sequence number it will be detected because it is covered by the ICV.

MACsec threats: MACsec does not duplicate user data frames.

Well-known security threats: Man-in-the-middle, man at the end recording frames and injecting frames at a later time, i.e., replay attack.

6.3.5 Frame transit delay

The MAC Service introduces a variable frame transit delay that is dependent on media types and media access control methods. Frame transit delay is the elapsed time between an MA_UNITDATA.request primitive and the corresponding MA_UNITDATA.indication primitive. Elapsed time values are calculated only on Service Data Units that are successfully transferred.

MACsec threats: Since the MAC Service is provided at an abstract interface within an end station, it is not possible to specify the total frame transit delay precisely. It is, however, possible to measure the media access and frame transmission and reception, and the transit delay introduced by an intermediate system, in this case a Bridge.

The exact amount of transit delay caused by MACsec is determined by the algorithms chosen and the particular hardware used.

6.3.6 Frame lifetime

The MAC Service mandates an upper bound to the transit delay experienced for a particular instance of communication. This maximum frame lifetime is necessary to ensure the correct operation of higher layer protocols.

<<What is the policy when there is a frame to transmit and there is no SA? Should the frame be buffered and then sent?, or should it be dropped? How long does it take to set up the SA? need to state clearly the effect on timing.

MACsec threats: MACsec introduces a non-deterministic delay, on the order of ...

Well-known security threats: man-in-the-middle attacker can record and replay.

6.3.7 Undetected frame error rate

<<necessary in the spec>>

The MAC Service introduces a very low undetected frame error rate in transmitted frames. Undetected errors are protected against by the use of an FCS that is appended to the frame by the MAC Sublayer of the source station prior to transmission, and checked by the destination station on reception.

It is necessary for a SecY to recalculate the FCS when .....

NOTE—Application of the techniques described in IEEE Std 802.1D Annex F (informative) allow an implementation to achieve an arbitrarily small increase in undetected frame error rate, even in cases where the data that is within the coverage of the FCS is changed.

MACsec provides an error check many times greater than the original FCS.

6.3.8 Maximum Service Data Unit Size

The Maximum Service Data Unit Size that can be supported by an IEEE 802 LAN varies with the MAC method and its associated parameters (speed, electrical characteristics, etc.). It may be constrained by the owner of the LAN. .....

Well-known security threats: The MSDU size offers another opportunity for attack. Additional bits could be added to the frame so that it exceeds the network maximum allowable size. MACsec, with the ICV, will prevent this, though it will itself increase the size.

7. Principles of Secure Network Operation

<<This clause provides a network wide view of the operation of security and its impact on the operation of the network as a whole. The latter is almost trivial for if MACsec is restricted to point to point (only two possible peers) 'single hop' (one LAN) operation. It becomes more complex if security associations subdivide, separate, or distinguish the connectivity that would otherwise be available, for example, by having (a) some of the stations attached to a shared media LAN participate in a secure association while others are excluded (b) having the stations grouped into a number of secure associations (c) separating some group-wise associations such as those used for network configuration protocols. It becomes much more complex if the connectivity assumed by some associations rests upon the connectivity provided by others, as is likely in multi-hop scenarios, and exceedingly necessary to keep clear if a combination of multiple associations and multi-hop are present. >><<Which variants of multi-hop are we doing? Should be treated in the network operation chapter.>>

This clause establishes the principles and a model of Secure Network operation. It specifies the context necessary to understand how

a) the operation of each MAC Security Entity (SecY) (Clause 8);
b) establishment of secure associations between peer SecYs;
c) the operation of the MAC Security Protocol (Clause 9) between those SecYs; and
d) the relationship between SecYs and the other entities that compose Systems attached to LANs (Clause 8); and

support, preserve, and maintain the quality and security (Clause 6) of the Secure MAC Service, including

e) operation of network configuration protocols.

NOTE 1—Unless explicitly stated the use of the term 'secure network' in this Standard refers to a Bridged Local Area Network in part of which MAC Security is active.

7.1 Secure Network Overview

The principal elements of Secure Network operation comprise:

a) data source integrity
b) replay protection

and may also optionally include:

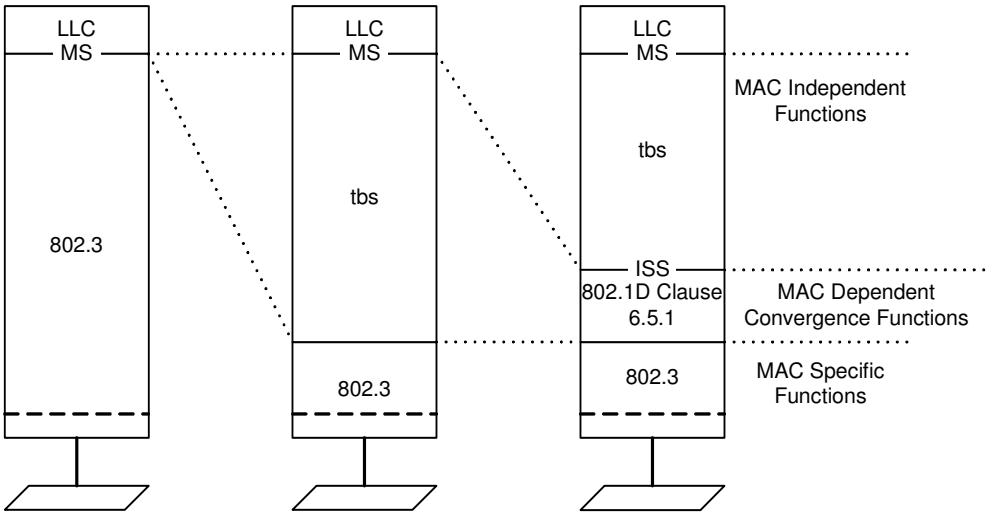
a) optional data privacy

They will not include:

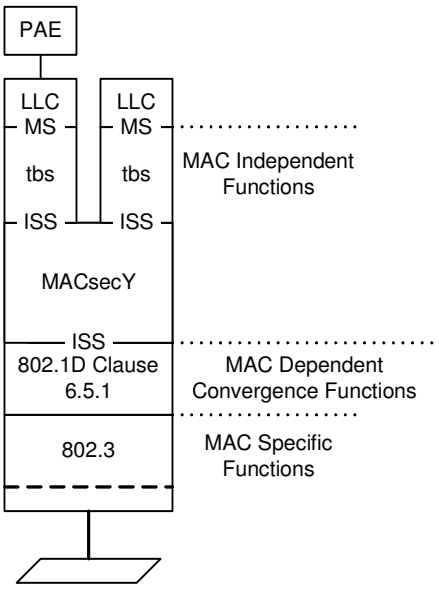
a) non-repudiation
b) protection of non-data packets

Non-data (control) packets are not included as they differ between various MACs and they cannot carry out their function if they are encrypted.

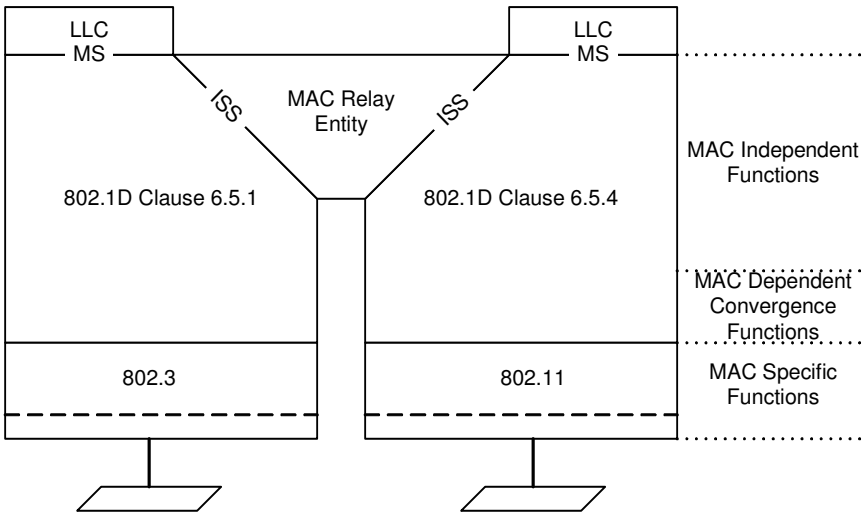
The position of the bridging function within the MAC Sublayer is shown in Figure.



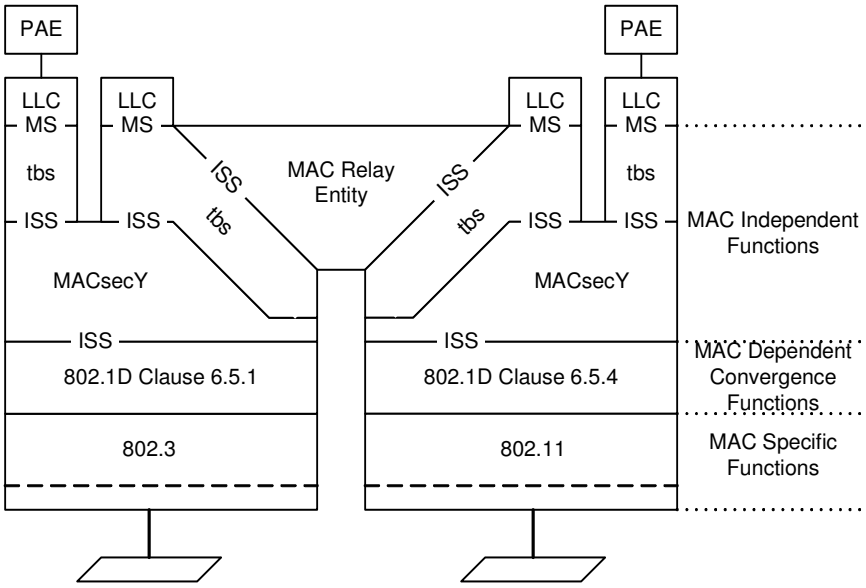
Three (equally valid) views of the MAC sublayer in an end station



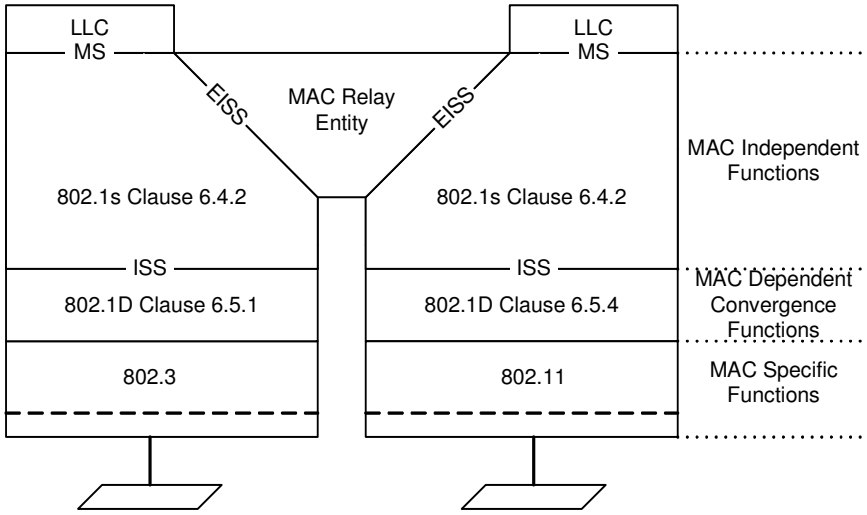
MAC sublayer in an end station with MAC Security



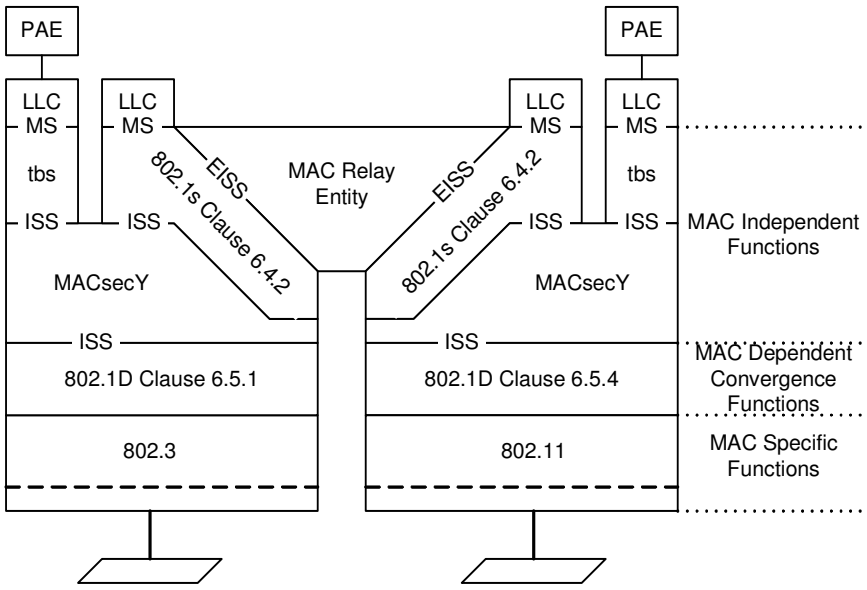
The MAC sublayer in an 802.1D MAC Bridge



The MAC sublayer in an 802.1D Bridge with MAC Security



The MAC sublayer in a VLAN-aware bridge



The MAC sublayer in a VLAN-aware bridge with MAC Security

8. Principles of MAC Security Entity operation

This clause

a) Introduces the principal elements of MAC Security Entity (SecY) operation (8.1) and the functions that support these elements.
b) Specifies an architecture (8.2), and a model (8.3) whose processes specify the detailed operation of the MAC Security Entity (8.4 thru 8.10).
c) Details the addressing requirements and specifies the addressing of SecYs (8.11).

8.1 SecY operation

Each SecY uses the MAC Service provided by a Common Port to provide one instance of a secured MAC Service, to the user of its Controlled Port, and one instance of an unsecured MAC Service, to the user of its Uncontrolled Port. The principal elements of SecY operation are

a) Secure transmission of MPDUs (MAC Protocol Data Units) from the Controlled Port (8.1.1).
b) Transparent transmission of MPDUs from the Uncontrolled Port (8.1.2).
c) Reception, verification, and delivery of secure MPDUs to the Controlled Port (8.1.3).
d) Reception and delivery of MPDUs to the Uncontrolled Port (8.1.4).
e) Support of administrative control over cipher suite selection (8.1.5)
f) Indication of available cipher suites to, and selection by, the Key Agreement Entity (KaY) (8.1.5).
g) Support of administrative control over the connectivity capabilities of the SecY (8.1.6).
h) Indication of connectivity capabilities to, and connectivity selection by, the KaY (8.1.6).
i) Support of administrative control over the optional security tagging capabilities of the SecY (8.1.7).
j) Transmission and reception key value updating, as requested by the KaY (8.1.8).
k) Indication of imminent transmission key value exhaustion, to the KaY (8.1.8).
l) Maintenance of MAC Status (8.13.2) and point to point parameters (8.13.3) for the Uncontrolled (8.1.9) and Controlled Ports (8.1.10).
m) Recording of events to facilitate management of the secured network and its components (8.1.11).

NOTE 1—The user of the Controlled Port can be another entity within the MAC Sublayer or an instance of LLC. The user of the Uncontrolled Port is an instance of LLC, providing one or more LSAPs (Link Service Access Points), each to an application or higher layer protocol.

NOTE 2—A local MACsec Key Agreement Entity (KaY) associated with each SecY negotiates, with a peer KaY or KaYs, which of the available cipher suites (8.4) is used by the SecY and its peer(s) to support secure communication.

NOTE 3—A ‘local entity’ resides within the same secure system and has a binding to the same Port as the SecY. Exchange of information with a local entity is modelled as occurring through a Layer Management Interface (LMI).

8.1.1 Secure Transmission

A SecY secures individual M_UNITDATA.requests made by the user of its Controlled Port. The functions that support secure transmission, through invocation of corresponding M_UNITDATA.requests to the provider of service at its Common Port, can include:

a) Encipherment of the user MAC Service Data Unit (MSDU) to provide *connectionless data confidentiality* (<ref to clause 6 security services>).
b) Calculation of an integrity check value (ICV) to protect the user MAC Protocol Data Unit (MPDU), and inclusion of the ICV in the MSDU of the Common Port M_UNITDATA.request, to provide *connectionless frame integrity* (<ref to definition of connectionless frame integrity in clause 3>, <ref to clause 6 security services>).

c) Inclusion of a secure origin authenticity (SOA) field comprising the individual MAC Address associated with the same Port as the SecY in the provider MSDU, to provide *secure origin authentication* (<ref to clause 6 security services>).

NOTE 1—The MAC Service Data Unit (MSDU) parameter accompanying an invocation of the MAC Internal Sublayer Service (ISS, 8.13.1) represents the data transmitted or received by the user of that service instance. The MAC Protocol Data Unit (MPDU) comprises the destination address, source address, and the data (MSDU) parameters of the invocation, but excludes the priority (8.12) and FCS parameters (8.13) whose values are local to the system and are not necessarily communicated unchanged to a peer user of the service.

NOTE 2—Except where explicitly specified otherwise, throughout this Clause the term "user" refers to the user of the MAC service instance provided by the Controlled Port, and the term "provider" refers to the instance of protocol and procedures that provides the MAC service instance to the SecY at the Common Port.

NOTE 3—The SOA field is not required on point to point links, as identified by the operPointToPointMAC status parameter of the service provider, as the secure association created by the PAE entity for the peer SecYs together with the direction of transmission of the secured MPDU can be used to identify the transmitting SecY.

The security algorithm used and its basic parameters compose a *cipher suite*. An identification of the selected cipher suite, together with the key values used by the cipher suite, and the other parameters used to determine the encoding, format, and presence of optional fields in provider MSDUs generated for a period by the SecY to provide secure MPDU transmission compose the *generation parameter set*. The corresponding parameters to be used by the intended recipient SecY or SecYs to decipher and verify the security of the received MPDUs compose the *verification parameter set*.

NOTE 2—The *verification parameter set* for a given MPDU can be the same as the *generation parameter set* or can differ if, for example, an asymmetrically keyed encryption algorithm is being used.

The SecY can include a Security TAG (SecTAG) in the initial octets of the provider MSDU, prior to the user data and ICV. The SecTAG comprises the MACsec Ethertype (<ref definition of Ethertype in Clause 3>, <ref allocation table in Clause 9>) and a Secure Association Identifier (SAID). The MACsec Ethertype identifies the provider MSDU as comprising the elements of a secured frame. The SAID identifies and is used by a receiving SecY to select the verification parameter set for the MPDU.

8.1.2 Transparent Transmission

For each M_UNITDATA.request made by the user of a SecYs Uncontrolled Port, the SecY makes an identical M_UNITDATA.request of its service provider.

8.1.3 Secure Reception

A SecY uses the currently selected cipher suite to verify the parameters of individual M_UNITDATA.indications from the Common Port.

If the octets immediately following the MACsec Ethertype comprise an SAID that identifies an acceptable verification set and the MPDU is verified, the SecY extracts the original MSDU and issues an M_UNITDATA.indication to the user of its Controlled Port.

<<add mention of management events/counts on discard?>>

8.1.4 Transparent Reception

For each M_UNITDATA.indication received from the Common Port, the receiving SecY issues an identical M_UNITDATA.indication to the user of its Uncontrolled Port. The relative order of provider indications and the corresponding indications to the users of the Uncontrolled Port and the Controlled Port is not defined, save that the order of indications from the provider to each of the Uncontrolled Port and the Controlled Port

are both independently preserved, and the interval between any provider indication and the SecYs corresponding indication to a user shall not exceed the bound specified in <ref performance parameters>.

8.1.5 Cipher suite selection

The security capabilities of a SecY are expressed in terms of its implementation of cipher suites. Each cipher suite represents a security algorithm together with the basic parameters of that algorithm. Cipher suites are further discussed in Clause <ref>.

A SecY shall be capable of using each of the following

a) the Null Cipher Suite ();
b) the Default Cipher Suite (CWC-AES), as specified in Clause (<ref-CWC-AES>);

and may implement

c) one or more additional MACsec standard Cipher Suites as identified in Table <8-Cipher Suites>, and specified by the relevant clause of this standard;
d) one or more proprietary Cipher Suites, as identified by the globally unique identifiers described in Clause <tbd>.

A Cipher Suite Selectable parameter is associated with each cipher suite, and can be set True or False by the system administrator. The MACsec Key Agreement Entity (KaY) associated with the Port is notified of the currently selectable Cipher Suites.

The KaY is responsible for selecting the Current Cipher Suite, and communicating that choice to the SecY via the LMI. If the Cipher Suite Selectable parameter for the Current Cipher Suite is set False, the SecY shall cease transmission and reception to and from the Controlled Port and set MAC_Operational (<ref>) False for that Port.

NOTE 1—Two communicating SecYs can implement a number of Cipher Suites in common. Specification of the process of selection from amongst the selectable cipher suites provided by a SecY is outside the scope of this standard, and is modeled occurring through the operation of a MACsec Key Agreement Entity (KaY). A user of services provided through the Controlled Port can retrieve parameters from the KaY to characterize both the Current Cipher Suite and the authorized capabilities of the users of the peer SecY(s). Selection of an inferior cipher suite can therefore result in restrictions in communication being imposed by that user.

NOTE 2—The Null Cipher Suite is included to facilitate interoperability in environments where other systems do not implement SecYs, without a more complex system level specification which would remove the SecY in such environments. The system administrator can clear the Cipher Suite Selectable parameter for the Null Cipher Suite to deny communication under these conditions. The Null Cipher Suite may also be used on connections that are secured by other means, such as physical security, but such use is not recommended.

8.1.6 SecY Connectivity

The connectivity capabilities of a SecY are expressed in terms of the number of other SecYs with which it is able to maintain secure communications. When the Current Cipher Suite is the Null Cipher Suite every SecY is capable of maintaining its Controlled Port in an unsecured association with an unidentified number of other systems, only limited by the capabilities provided by the Common Port.

When using a cipher suite other than the Null Cipher Suite, a SecY shall be capable of

a) supporting a secured point-to-point association with one other SecY;

and may be capable of

b) supporting a secured many-to-many association with a specified maximum number of other SecYs.

NOTE 1—The specification of a MACsec Key Agreement Protocol capable of supporting many-to-many associations exceeds our present ambitions.

The connectivity of a SecY can be managed, independently for each cipher suite, by the Admin Multipoint Connectivity variable. This variable is always True for the Null Cipher Suite. For all other cipher suites the default is False, and the value may be changed if the cipher suite supports many-to-many associations.

The MACsec Key Agreement Entity (KaY) associated with the Port is notified of the value of the Admin Multipoint Connectivity variable for each of the currently selectable cipher suites. The KaY is responsible for selecting and communicating the value of the current cipher suite's Oper Multipoint Connectivity variable to the SecY via the LMI.

NOTE—It is not expected that the KaY use the Cipher Suite Admin Multipoint Connectivity to preferentially negotiate the Current Cipher Suite with a peer SecY or SecYs. However cipher suite selection can constrain connectivity.

8.1.7 Security tagging options

The security tagging capabilities of a SecY reflect its ability to prepend a Security TAG (SecTAG) to user data requests from the Controlled Port, and to remove and act upon the SecTAG on receipt of data indications from the Common Port. The format of SecTAG is specified in Clause 9, and comprises a MACSec Ethertype followed by an SAID (<ref>) and an optional SOA (<ref>).

Every cipher suite other than the Null Cipher suite provides confidentiality, through data encryption, or integrity, by calculating and appending an ICV to the user data, or both confidentiality and integrity, and requires the addition of a SecTAG with an SAID.

When the Current Cipher Suite is the Null Cipher Suite, every SecY is capable of transmitting frames to and from the Controlled Port without the addition or removal of a SecTAG, and the SecTAG is omitted by default.

An Include TAG variable shall be implemented for each cipher suite including the Null Cipher Suite. This variable is always True for all cipher suites other than the Null Cipher Suite, for which it is False by default. A SecTAG shall be included in a frame transmitted from the Controlled Port when the Null Cipher Suite is being used if, and only if, the Include TAG and the Neighbors All SecYs variables (see below) are both True.

A SecY shall be capable of including an SOA in the SecTAG. An Include SOA variable shall be implemented for each cipher suite including the Null Cipher Suite, and takes one of the following values

a) Always, the SOA is included whenever the SecTAG is included
b) Multipoint Only, the SOA is included only if the SecTAG is included and Oper Multipoint Connectivity is True

The tagging options for each cipher suite are not communicated to the KaY and play no role in cipher suite selection, however if the KaY selects the Null Cipher Suite but has determined that all stations connected to the LAN include SecYs, it can set the Neighbors All SecYs parameters for the SecY.

If the Current Cipher Suite is the Null Cipher Suite, the SecY processes all received frames whether or not they contain an SecTAG. If the Current Cipher Suite is not the Null Cipher Suite and Oper Multipoint Connectivity is False, the SecY processes all received frames containing an SAID whether or not they contain a SOA.

8.1.8 Cipher suite keying

Keys for the Current Cipher Suite are supplied and periodically updated by the MACsec Key Agreement Entity associated with the SecY. The SecY may provide an indication of probable imminent key exhaustion to prompt agreement of a new key.

8.1.9 Uncontrolled Port MAC status and point to point parameters

The value of the MAC_Enabled (8.13.2) and MAC_Operational parameters (8.13.3) for the Uncontrolled Port shall be the same as that for the Common Port.

If the adminPointtoPointMAC parameter (8.13.3) for the Uncontrolled Port has the value Auto, then the value of the operPointtoPointMAC parameter shall be the same as that for the Common Port.

8.1.10 Controlled Port MAC status and point to point parameters

The value of the MAC_Enabled (8.13.2) and MAC_Operational parameters (8.13.3) for the Uncontrolled Port shall be the same as that for the Common Port.

<<Clearly wrong, it is unclear what MAC_Enabled should reflect.>>

If the value of the adminPointtoPointMAC parameter associated with the Controlled Port is Auto, then the value of operPointtoPointMAC parameter shall be

a) the same as that for the Common Port, if the Current Cipher Suite is the Null Cipher Suite; and
b) the value of the Current Cipher Suite Multipoint Connectivity variable otherwise

8.1.11 MACsec management

The functions that support MACsec management control and monitor the provision of the above functions. They are specified in Clause?.

8.2 SecY architecture

A SecY uses an instance of the MAC Internal Sublayer Service (ISS, 8.13), referred to as the Common Port, to provide a secured instance of the ISS, the Controlled Port, and an unsecured instance of the ISS, the Uncontrolled Port, that provides transparent transmission and reception through the Common Port. The architecture of a SecY is illustrated in Figure 8-1, and comprises:

a) The Controlled, Uncontrolled, and Common Ports together with their MAC Status parameters;
b) One or more Cipher Suite Implementations, only one of which can be in operation at a time;
c) A Transmit Mux, which supports interleaved transmission, on an ISS service request by service request basis, both by the Uncontrolled Port and by the processes that support transmission by the Controlled Port;
d) A Receive Demux, which provides the received parameters of each service indication from the Common Port to the Uncontrolled Port, and to the processes that support reception by the Controlled Port;
e) A Transmit Encoder, which encodes (as required), a SecTAG (<defn. ref>), an SAID (Secure Association Identifier, <defn. ref>), an optional SOA (Secure Origin Authenticity, <defn. ref>), the encrypted MAC Service Data Unit (MSDU) of an Controlled Port data request, user data, and an ICV (Integrity Check Value, <defn. ref>), into the MSDU that accompanies the corresponding Common Port data request.

f) A Receive Decoder, which examines the parameters of the MAC Service Data Unit (MSDU) of each receive data indication, extracting (if present) the SAID, the encrypted data, and the ICV, and presents these parameters to the Cipher Suite Implementation.

g) An optional transmit and receive FCS Regenerators ().

A Layer Management Interface (LMI) is used by the processes that compose the SecY to communicate the capabilities of the SecY, its status, and protocol and management events and counters to other Entities that compose the secure system of which the SecY forms a part, and to receive parameters from those Entities. Specifically, the LMI is used:

h) To allow a MACsec Key Agreement Entity (KaY) responsible for the SecY to discover the Cipher Suites implemented by the SecY, prior to cipher suite negotiation and selection with a peer KaY forming part of a system incorporating a peer SecY.

i) To receive cipher suite selection and Generation and Verification Parameter Sets (keying material) from the KaY, and to request new keying material prior to exhaustion of that currently in use.

8.3 Model of operation

The model of operation is simply a basis for describing the functionality of a SecY. It is in no way intended to constrain real implementations; these may adopt any internal model of operation compatible with the externally visible behavior that this standard specifies. Conformance of equipment to this standard is purely in respect of observable protocol.

8.4 Cipher Suite Implementation

The Null Cipher Suite simply passes each Controlled Port M_UNITDATA.request to the Common Port, and each Common Port M_UNITDATA.indication to the Controlled Port without changing any of the parameters of each request or indication.

While the detailed specification of the operation of each cipher suite is contained in the appropriate clause of this standard, each provides (as applicable) confidentiality and integrity services for the same M_UNITDATA.request and M_UNITDATA.indication parameters.

Where confidentiality is provided, the entirety of the user data parameter (MSDU) supplied in the corresponding Controlled Port M_UNITDATA.request is encrypted, as illustrated in Figure 8-1. The other parameters of the request, i.e. the destination address, source address, and priority are not encrypted. The fields of the SecTAG, i.e. the MACsec Ethertype, SAID, and optional SOA are not encrypted.

Where data integrity is provided, an Integrity Check Value (ICV) is calculated that provides integrity for the destination address, source address, user data (after encryption, if applicable). The ICV does not include the M_UNITDATA.request priority parameter nor the fields of the SecTAG (MACsec Ethertype, SAID, and optional SOA). See Figure 8-1.

<<Should the ICV cover the SecTAG fields. Not as described here, nor as shown in Figure 8-1, but may be a good idea. Pluses and minuses to discuss.>>

<< describe: Figure 8-1 just an indication of functionality, no need for separate Integrity Check and Encryption/Decryption with some algorithms, purpose of identifying a “cipher suite”, cipher suite Table, performance issues particularly “stuttering” after missed frames, “parameter sets” in more detail - possibly substituting a “sec. head” name, when indications occur and when they don’t (ICV or decrypt failure), perhaps some management diagnostic counts (or a reference note)>>

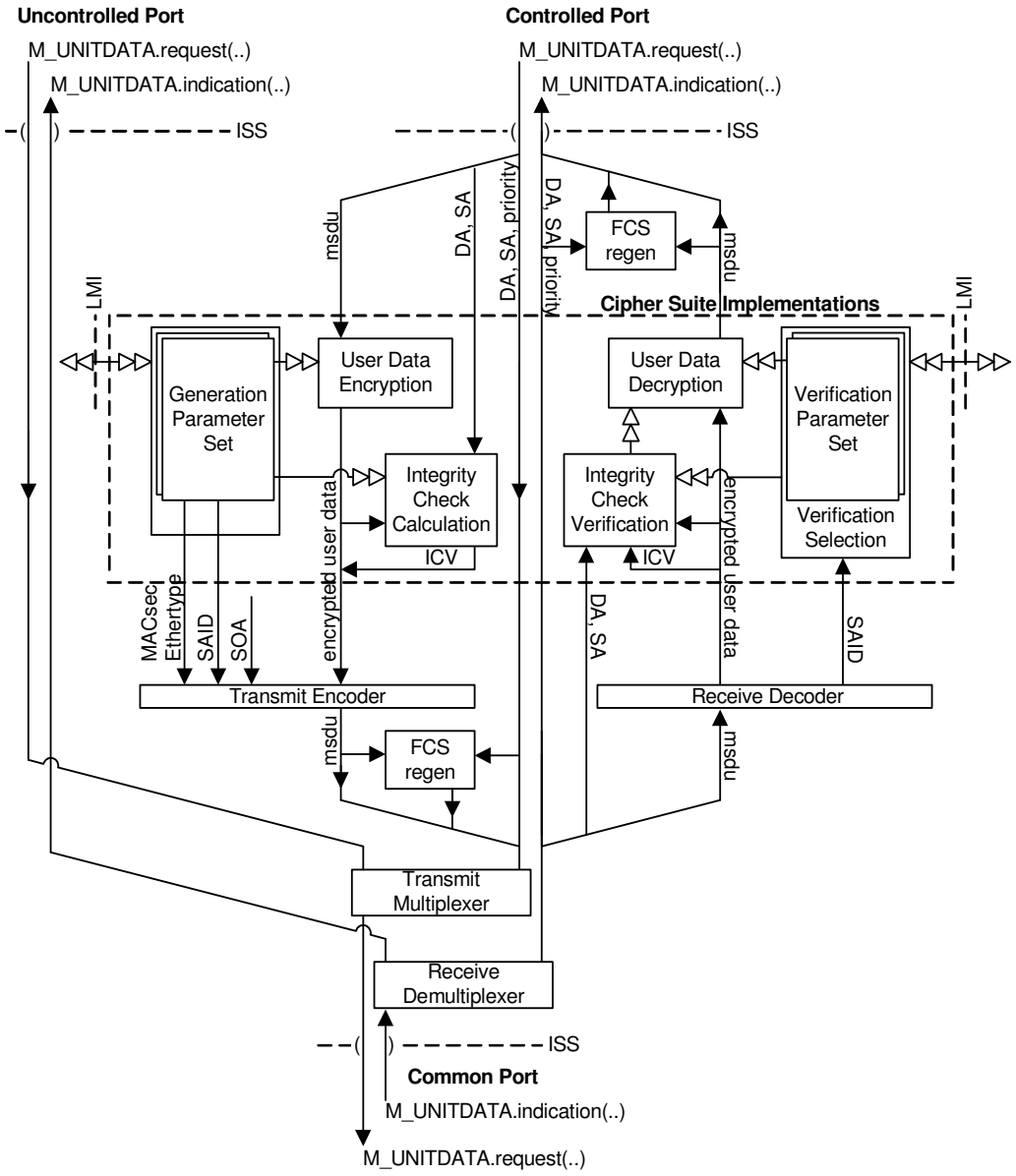


Figure 8-1—SecY Architecture and Operation

8.5 Transmit Multiplexer

The Transmit Multiplexer accepts data requests from the Uncontrolled Port, and the parameters of data requests from the Controlled Port and the cipher suite implementation and Transmit Encoder supporting that Port, and makes corresponding data requests of the Common Port.

The relative order of requests to the Common Port provider and the corresponding user requests at the Uncontrolled Port and at the Controlled Port is not defined, save that the order of requests from each of the Uncontrolled Port and the Controlled Port are both independently preserved, and the interval between any user request and the SecYs corresponding request of the provider shall not exceed the bound specified in <ref performance parameters>.

8.6 Receive Demultiplexer

Each M_UNITDATA.indication from the Common Port is submitted to the Receive Decoder for the Controlled Port and to the Uncontrolled Port.

NOTE—This specification most clearly sets out the resulting behavior of a conforming implementation. Real implementations can implement the behavior in any way that yields the same externally visible behavior, including the values of management counters, as described in clause 8.3. For example, examination of the specification in this Clause shows that there need be no implementation burden corresponding to duplication of the received frame if the Current Cipher Suite is not the Null Cipher Suite and none of the users of the LLC Entity supported by the Uncontrolled Port make use of the MACsec Ethertype.

8.7 Transmit Encoding

Successive octets of the MSDU that will accompany the Common Port M_UNITDATA.request corresponding to the original user request at the Controlled Port are used to encode the MACsec Ethertype, the SAID representing the generation parameters used by the cipher suite, the SOA, and the Controlled Port MSDU (possibly encrypted), and the ICV, subject to the following

a) If the Current Cipher Suite is the Null Cipher Suite and either the Include TAG or the Neighbors All SecYs variable is False, then the MACsec Ethertype, SAID, and SOA are omitted.
b) If the Include SOA variable has the value Multipoint Only and Oper Multipoint Connectivity is False, then the SOA is omitted.
c) If the Current Cipher Suite is the Null Cipher Suite, no ICV is present.

NOTE—The length of the ICV is dependent on the cipher suite used, and is transparent to the operation of the Transmit Encoding and Receive Decoding processes.

The addition of the SecTAG parameters, encryption of the user data, and addition of the ICV to the MSDU submitted to the Common Port ensure that the maximum size of MSDU required from the Common Port exceeds that provided at the Controlled Port. If, following transmit encoding, the MSDU to be supplied as a parameter of a data request at the Controlled Port exceeds the maximum provided, the frame is discarded and the Oversized Encoded Frame count incremented. Details of the discarded frame, similar to those specified in 802.1D's "Discard On Error Details" structure may be recorded to assist network management resolution of the problem.

<<NOTE—Discussions have been held with the leadership of 802.3 as to the need to increase the permissible frame size specifically to allow the 'normal maximum' user data to be carried in a single frame. When this draft standard has sufficiently advanced for a definite estimate of the additional octets required that discussion will be refreshed and formal permission for the increase sought from 802.3.>>

8.8 Receive Decoding

If the initial octets of the MSDU accompanying a Common Port M_UNITDATA.indication compose the MACsec Ethertype, the SAID and the SOA (if present) are extracted from the MSDU.

If the currently selected cipher suite is the Null Cipher Suite, the remaining octets of the MSDU compose the MSDU parameter of the corresponding Controlled Port M_UNITDATA.indication.

If the SAID is not present, and the currently selected cipher suite is not the Null Cipher Suite, the received frame is discarded.

If both the SAID and the SOA are required to uniquely identify the verification parameter set for the received frame, i.e. the current cipher suite's Oper Multipoint Connectivity variable is set, and the SOA is not present, the received frame is discarded.

<<Whether both SAID and SOA are really required for Multipoint Connectivity depends on what we determine to be included in the SAID. Perhaps we should define a SAID to comprise an SOA and a SEQN, with the SOA potentially absent for point to point connectivity? That would make the specification easier to write.>>

Otherwise the SAID and SOA (if present) are presented to current cipher suite implementation to facilitate retrieval of the verification parameter set, together with the remaining octets of the provider MSDU, comprising the (possibly encrypted) peer user data and ICV.

NOTE 1—The short phrase "the frame is discarded" is commonly used to express the more formal notion of not processing a service primitive (an indication or request) further and recovering the resources that embody the parameters of that service primitive. If a duplicate of the primitive has been submitted to another process, by the Receive Demultiplexer in this case, processing of that duplicate is unaffected.

<<Add management counters incremented by the Receive Decoding process.>>

8.9 Transmit and Receive FCS Regenerators

A frame check sequence (FCS) can be included as a parameter of an M_UNITDATA.request or M_UNITDATA.indication primitive. When the data that is within the FCS coverage is modified, by the addition of an integrity check value (ICV), or encryption of the user data, the FCS changes. The SecY shall not introduce an undetected frame error rate greater than that which would have been achieved by preserving the original FCS.

NOTE—There are number of possibilities for changing FCS without diminishing the coverage provided. One is to generate a new FCS by algorithmically modifying the received FCS, based on knowledge of the FCS algorithm and the transformations that the frame has undergone between reception and transmission.

8.10 MACsec Key Agreement Layer Management Interface

<<Sum up all the parameters shuffled to and from the KaY in this clause>>

8.11 Addressing

Frames transmitted between end stations using the MAC Service carry the MAC Address of the source and destination peer end stations in the source and destination address fields of the frames, respectively. Communicating peer SecYs can secure communication for all or part of the path used by such frames, and are not directly addressed by the communicating peers, nor are the frames modified to include additional addresses. Each SecY does not have a MAC Address of its own account, but is associated with a local entity that forms part of the secure system and is responsible for discovering potential peer SecYs and enabling MACsec Key Agreement, including cipher suite selection, to facilitate communication.

The addressing used by Key Agreement Entities and the means they use to identify SecYs within the same secure system are outside the scope of this specification.

Each system that includes a SecY has at least one MAC Address that can be used for the purpose of generating globally unique SAIDs and SAOs, if these are required.

While destination and source MAC addresses are not required to identify SecYs, they are parameters of the MAC Internal Sublayer Service (ISS) used and provided by a SecY, and can be covered by the ICV

(Integrity Check Value) generated by a Cipher Suite Implementation while remaining unencrypted. To facilitate ICV calculation and verification all frames processed by SecYs shall use 48-bit MAC addresses.

8.12 Priority

While priority is a parameter of both an ISS M_UNITDATA.request and corresponding M_UNITDATA.indications, end to end communication of the requested priority is not a service attribute (<ref clause 6>). Protocols supporting the ISS can use the requested priority to perform local actions in the originating station, and do not necessarily attempt to communicate the parameter. Accordingly, the requested and indicated priorities do not contribute to the ICV, and are not explicitly included in the encoded MSDU by a transmitting SecY.

NOTE—If communication of priority is desired, either guaranteed unchanged or available to a service provider for possible modification to meet the admission control and service characteristics of a particular network, use of the EISS in conjunction with the ISS is indicated. See Clause 7, Principle of Network Operation.

8.13 Internal Sublayer Service

The Internal Sublayer Service (ISS) augments the specification of the MAC Service (ISO/IEC 15802-1) with elements necessary to the performance of the relay function. Within an end station, these additional elements are considered to be either below the MAC Service boundary, and pertinent only to the operation of the service provider; or local matters not forming part of the peer-to-peer nature of the MAC Service. The ISS excludes MAC-specific features and procedures whose operation is confined to an individual LAN.

NOTE 1—No new service primitives are defined. The frame_check_sequence is added to list of parameters associated with the MA_UNITDATA.request and MA_UNITDATA.indication primitives.

8.13.1 Service primitives and parameters

The ISS is specified by two unit-data primitives, an M_UNITDATA.indication and an M_UNITDATA.request, together with the parameters of those primitives. Each M_UNITDATA indication corresponds to the receipt of an error-free MAC frame from a LAN. A data request primitive is invoked to transmit a frame to an individual LAN.

NOTE 1—Detailed specifications of error conditions in received frames are contained in the relevant MAC standards; for example, FCS errors, length errors, non-integral number of octets.

```
M_UNITDATA.indication    (
                         destination_address,
                         source_address,
                         mac_service_data_unit,
                         priority,
                         frame_check_sequence
                         )

M_UNITDATA.request       (
                         destination_address,
                         source_address,
                         mac_service_data_unit,
                         priority,
                         frame_check_sequence
                         )
```

The **destination_address** parameter is the address of an individual MAC entity or a group of MAC entities. The **source_address** parameter is the individual address of the source MAC entity. The **mac_service_data_unit** parameter is the service user data. The default **priority** value is 0. Values 1 through 7 form an ordered sequence of priorities, with 1 being the lowest value and 7 the highest.

The **frame_check_sequence** parameter may be provided so that it can be used in a related primitive. The parameter comprises the FCS value and sufficient information to determine whether the FCS value can be used.

The identification of the LAN or other service instance from which particular frames are received is a local matter and is not expressed as a parameter of the service primitive.

NOTE 3—The ISS specification in this standard differs from that in IEEE Std 802.1D-2003 as it omits the frame_type and access_priority parameters. The frame_type is not required as the receipt of a frames other than a user data frame does not cause a data indication, nor are such frames transmitted by the media independent bridge functions. The mapping of the ISS to particular access methods specified by this standard includes derivation of the access_priority parameter (for those media that require it) from the ISS priority parameter.

8.13.2 Status parameters

The Internal Sublayer Service also makes available status parameters that reflect the operational state and administrative controls over each instance of the service provided.

The **MAC_Enabled** parameter is TRUE if use of the service is permitted; and is otherwise FALSE. The value of this parameter is determined by administrative controls specific to the entity providing the service, as specified in [6.5].

The **MAC_Operational** parameter is TRUE if the entity providing the service is capable of transmitting and receiving frames and its use is permitted by management, i.e. MAC_Enabled is also TRUE. Its value is otherwise FALSE. The value of this parameter is determined by the specific MAC procedures, as specified in [6.5.]

NOTE—These status parameters provide a common approach across MACs for handling the fact that:
a) A MAC can inherently be working or not;
b) If the MAC is working, its operational state can be administratively overridden.

8.13.3 Point-to-point parameters

The Internal Sublayer Service also makes available status parameters that reflect the point-to-point status of each instance of the service provided and provide administrative control over the use of that information.

If the **operPointToPointMAC** parameter is TRUE if the service is used as if it provides connectivity to at most one other system, if FALSE the service is used as if it can provide connectivity to a number of systems.

The **adminPointToPointMAC** parameter can take one of three values. If it is

a) **ForceTrue**, operPointToPointMAC shall be TRUE, regardless of any indications to the contrary generated by the service providing entity.
b) **ForceFalse**, operPointToPointMAC shall be FALSE.
c) **Auto**. operPointToPointMAC is determined by the service providing entity, as specified in.

The value of operPointToPointMAC is determined dynamically; i.e., it is re-evaluated whenever adminPointToPointMAC or the status of the service providing entity changes.

9. MAC Security Protocol (MACsec)

MACsec provides security services on a frame by frame basis, using cryptographic methods within the context of security associations maintained by MACsec Key Agreeement.

This clause:

a) Sets out design requirements for the MAC Security Protocol (MACsec) (9.1);
b) Specifies the procedures and elements of the MAC Security Protocol (MACsec) (9.2);
c) Specifies the generation and validation of protocol elements in terms of the operation of a Cipher Suite, and relates each of the protocol parameters to the terms usually used in the specification of a Cipher Suite;
d) Describes the criteria adopted during the development of this standard for the inclusion of standard Cipher Suites, and makes recommendations on the application of these criteria to the use of additional Cipher Suites;
e) Describes the use of Security Associations to select the Cipher Suite to be used and to provide keys and other parameters necessary to the operation of the Cipher Suite;
f) Specifies the requirements that MACsec places on the operation of Key Agreement protocols in their selection of the Cipher Suite to be used and the establishment of Security Associations, to ensure successful service provision by MACsec.

<< sort out (e) and (f) as to who is doing what and who, like an SA, is just a name for a consequence.>>

NOTE—The operation of MACsec Key Agreement, and the selection of protocols it uses to establish Security Associations are outside the scope of this standard and the subject of an ongoing standards project. However the resulting connectivity and parameters provided to the MAC Security Entities are essential to the operation of MACsec.

9.1 Protocol design requirements

MACsec operates in Bridged Local Area Networks and Virtually Bridged Local Area Networks comprising individual point to point or shared media LANs arbitrarily interconnected by MAC Bridges and VLAN Bridges. Each of the end systems and Bridges may incorporate MAC Security Entities (SecYs). MACsec supports, preserves, and maintains the quality of the Secure MAC Service in all its aspects as specified by Cl 6, meeting requirements for

a) Connectivity (9.1.1)
b) Interoperability (9.1.2)
c) Scalability (9.1.4)
d) Validity of security claims (9.1.3)

These requirements are met in part by the architecture that specifies how MAC Security Enties are placed within LAN stations and communicate with selected peers, as described in Clause 7 and summarized below; in part by the operation of MACsec directly, as described in Clause 9.2; in part by placing requirements for support of the protocol on each MAC Security Entity and the system that contains it, as described in Clause 9.4; in part by the choice of cryptographic methods that compose each MACsec cipher suite, as described in Clauses 9.3 and 9.6; and in part by requirements placed on the operation of the protocols that support MACsec Key Agreement, including aspects of authentication, authorization, and distribution of keys, as described in Clause 9.4.

9.1.1 Connectivity Requirements

The design of MACsec ensures that the protocols that configure, and that run over, media, individual LANs, and Bridged or Virtual Bridged Local Area Networks as a whole, can continue to operate with no dimunition in the capabilities available to and customarily used by network administrators.

<<No strange new threads of connectivity appear, LANs continue to look like LANs and are secured as individual LANs, data and control connectivity remain tied together, bridge and firewall filters can still be applied to specific protocols (with some improvements depending on MACsec Key Agreement for authorization support), perhaps the place to throw in a teaser reference to key agreement protocols capable of resource chaining, some introduction to the following bullets which express some of the foregoing - with the introduction of timing bounds and aggregate limits on departures from the expressed goals, when these are allowed and how to fix in the following paragraphs which outline how these requirements are met.>>

e) Connectivity is continuous. Even though connectivity is support by a series of Security Associations whose lifetime is relatively brief compared to the uninterrupted connectivity expected from MACsec.
f) Connectivity is symmetric, i.e., if system A can communicate with system B, B can communicate with A.
g) Connectivity is transitive, i.e., if system A can use the secure MAC service at one MSAP to communicate with system B and system C then system B and system C can communicate using their same MSAPs.

<<outline how these requirements are met; need to interrupt connectivity at at least some of participants when “LANs merge”>>

9.1.2 Interoperability Requirements

<<Interoperability and deployability, need for the Default Cipher Suite, limiting choices through cipher suites and specifying a minimum number to cover the combinations of security required as mandatory, crucial need to avoid or minimise startup >>

9.1.3 Validity Requirements

<<ensured through cipher suite selection, and more>>

9.1.4 Scalability Requirements

<<Some introduction to the following. Also references to other aspects of resource usage. Number of active SAs required per direction of communication from a port limited to 2 (how the Provider Bridge scenario is not an exception to this rule)>>

a) The resources required to support MACsec in any single LAN station (an end station or a Bridge Port) are independent of the total number of systems that compose the network.
   This requirement is met by ensuring that each MAC Security Entity provides a secure service (the secure Internal Sublayer Service, ISS, <ref-clause 8>) to peer MAC Security Entities attached to the same LAN, and not to arbitrary stations throughout the network.

<< and all the undesirable things that might cause scaling failure, that someone might anticipate might cause SecY requirements to go up.>>

9.2 MAC Security Protocol (MACsec) Operation

MACsec comprises modification and additions to the user data (MAC Service Data Unit, MSDU) conveyed by each frame transmitted by a user of the protocol. No additional frames are introduced by the operation of the protocol, and each frame is delivered unmodified to peer users, subject to secure validation of the origin, destination and source address, and user data conveyed.

The encoding and decoding of the MSDU in and from a MACsec MAC Protocol Data Unit (MPDU) that accompanies a request (or indication) to (or from) the MAC service provider used by a MAC Security Entity (to communicate with its peers) is illustrated in Figure 9-1.

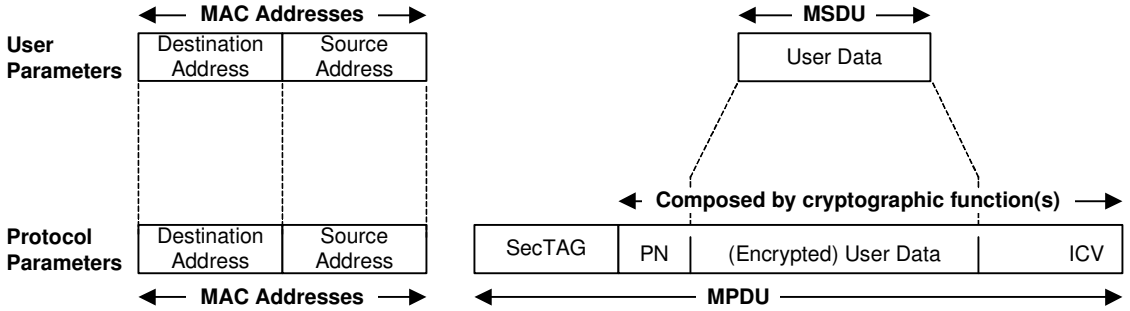


Figure 9-1—SecY Architecture and Operation

Each cryptographic cipher suite implementation calculates an Integrity Check Value (ICV), or an equivalent parameter, that ensures the integrity of the MAC Destination Address, MAC Source Address, and User Data, and is included within the octets that compose the cryptographic component of the MPDU. The size and format of the ICV, or its equivalent, are particular to the cipher suite and is specified in the relevant clause for each Cipher Suite (see Table 9-1)

A cryptographic cipher suite may also specify encryption of the User Data (but not the destination or source addresses). A packet number (PN) field is typically used to vary the initial state of the cryptographic calculation prior to encoding user data, and forms part of the cryptographic component of the MSDU, as illustrated in Figure 9-1. The length of the PN is either 24 bits or 64 bits, as specified by the relevant clause for the Cipher Suite in use.

NOTE—The term “cryptographic cipher suite” is consistently used to refer to all cipher suites other than the Null Cipher Suite. While the latter is a necessity to allow systems incorporating MAC Security to be deployed in insecure or transitional environments, it often proves the exception in a description of cipher suite behavior. Note that secure operation can be ensured by administratively disabling the use of the Null Cipher Suite, using the management controls specified in Clause 8.

The Security TAG (SecTAG) comprises the following fields:

a) A MACsec EtherType, that serves to distinguish MACsec protocol frames from other frames where both exist on the same medium;
b) An Security Association Identifier (SAID,) that serves to identify the cipher suite and key value(s) to be used to decrypt and validate the parameters of a received frame;
c) An optional Security Origin Authenticity (SOA); that serves to identify the MAC Security Entity that encoded the cryptographic component of the MACsec MSDU.

The format of the Security TAG is specified in Clause 10, and is independent of the cipher suite used.

9.2.1 Protocol transmission

The SecTAG can be omitted on transmission if the Null Cipher Suite is being used. Otherwise the SecTAG shall convey the SAID to be used by the intended MAC Security Entity peer recipient(s) of the transmitted frame, and can include the SOA. Management controls relevant to transmission are described in Clause 8, and are fomally specified by the state machines and accompanying procedures in this clause (<ref-tbs>).

<<must say what happens when PN space exhausted, just have to stop transmitting, think this is clear now, no fallback possible at least within the SecY, Key Agreement may select Default Cipher Suite in systems where SecY user is capable of invoking policies based on Key Agreement’s provided level of assurance, but such operation is outside and does not alter the specification/operation of a SecY. Mention signal to Key Agreement as SAID gets retired (on what criteria, or is there only one SAID at a time with very brief overlap on transmit) and separate signals as (a) PN is close to running out (b) Key Agreement has run out, model as

status varaiable setting not active event to avoid concerns and elaborate design of state machines to avoid “load”.>>

<<Following needs expanding a treating slightly differently.>>

The additional size added to the IEEE 802.3 header exceeds the hardware limitations of some IEEE 802.3 implementations, requiring some adjustment. fragmentation. A requrest for larger frame size has been made for consideration to 802.3.

9.2.2 Protocol reception

If a received MPDU contains at least two octets, and the first two octets compose the MACsec Ethertype, then the SecTAG is deemed to be present and is validated as specified in Clause 10. If the SecTAG fails the specified validation checks then the received frame is discarded, and no indication is made to the user of the MAC Security Entity.

If the Current Cipher Suite is the Null Cipher Suite, and the SecTAG is either present and valid or is absent, a receive indication is made to the MAC Security Entity’s user, with the remaining octets of the MPDU (i.e. those following the SecTAG, if present) composing the MSDU parameter of that indication.

If the Current Cipher Suite is not the Null Cipher Suite, and a valid SecTAG is not present, the received frame is discarded, and no indication is made to the user of the MAC Security Entity.

If the Current Cipher Suite is not the Null Cipher Suite and a valid SecTAG is present, the SAID is extracted from the SecTAG and presented to the cipher suite implementation together with the destination address, source address, and the octets of the MPDU that follow the SecTAG. The cipher suite implementation validates the addresses and remaining octets (i.e. applies an integrity check to ensure they remain as originally transmitted by the peer MAC Security Entity), and extracts the original peer MSDU. If the SAID is invalid, or the validation or MSDU extraction fails, the received frame is discarded, and no indication is made to the user of the MAC Security Entity. Otherwise a receive indication is made to the MAC Security Entity’s user, with the received addresses and the extracted MSDU as parameters of that indication.

Management controls and recording of the discard events are described in Clause 8, and are formally specified by the state machines and accompanying procedures in this clause (<ref-tbs>).

9.3 MACsec Cipher Suites

A MACsec Cipher Suite specifies a set of algorithms that provide MAC Security services, and the parameters (for example, key size) to be used with those algorithms. The choice and combination of cryptographic methods is notorious for the introduction of unexpected security exposures, and this standard only permits the use of cipher suites that meet well defined criteria (9.6). Specification of the cryptographic functions required by MAC Security in terms of cipher suites increases interoperability by enabling the specification of a clear default and a very limited number of alternatives.

An implementation of MAC Security that claims full conformance to this standard shall implement the Mandatory Cipher Suites in Table 9-1, may implement one or more of the Optional Cipher Suites in the Table, and shall not implement any other Cipher Suite, or other combination of cryptographic algorithms and parameters.

NOTE 1—Other cryptographic algorithms can be implemented, and are indeed expected to be implemented to support MACsec Key Agreement, but shall not be used to support the MAC Security protocol as specified in this Standard.

An implementation of MAC Security that implements the Mandatory Cipher Suites in Table 9-1 and one or more other Cipher Suites not in the Table, but in full conformance with the requirements of Clause 9.6 may claim <weasel> conformance with this Standard.

Table 9-1 assigns a cipher suite reference number for use in protocol identification within a MACsec context, provides a short name for use in this standard, indicates the type of cryptographic algorithm used and the security services provided, specifies whether the cipher suite is mandatory or optional for conformance to this standard, and references the clause of this standard that provides the defintive description of the cipher suite. The selection of both mandatory and optional cipher suites for inclusion in this standard conforms to the criteria of Clause 9.6, further information on the choice of the mandatory cipher suites is provided in Annex ?.

Table 9-1—MACsec Cipher Suites

Cipher Suite #	Name	Type	Services provided			Mandatory/ Optional	Defining Clause
			Integrity	Confidentiality	Replay protection		
0	Null Cipher Suite	Identity transform	No	No	No	Mandatory	9
1	Default Cipher Suite	CWC-AES	Yes	Yes	Yes	Mandatory	11?
2	Default Integrity Suite	PMAC	Yes	No	No	Mandatory	12?
3		AES-128 in CCM Mode	Yes	Yes	Yes	Optional	13?
4		AES-128 in OCB Mode	Yes	Yes	Yes	Optional	14?
5		OMAC	Yes	No	No	Optional	15?

NOTE—While a further clause of this standard provides the definitive specification of each cipher suite, that specificaton makes the greatest possible use of other public and established standards, and is principally concerned with ensuring unambiguous application of those standards in the context of MAC Security.

Certain elements are common to all the cryptographic Cipher Suites used by MAC Security:

a) Encryption is applied to the user data, and not to other fields supplied by the user of MAC Security.

b) The destination MAC address, source MAC address, and user data, and no other fields, are included within the scope of the integrity protection.

c) Where the specification of the cipher suite makes use of the term Additional Authenticated Data (AAD), the destination MAC address and source MAC address are supplied as AAD and compose a string of 12 octets, the first 6 octets representing the destination address in Canonical Format order (IEEE Std 802) and the next 6 octets representing the source address, also in Canonical Format order.

d) Where integrity is provided through the calculation of an ICV (Integrity Check Value) and the inclusion of an ICV in the cryptographic component of the MPDU, the ICV comprises the last 8 octets of the MPDU.

NOTE 1—Additional Authenticated Data (AAD) is also known as Additional Encrypted and Authenticated Data AEAD in some cipher suite specifications. <<Sure this is not right, did the original heading mean that some cipher suites have to duplicate AEAD in the encrypted portion for the cipher suite to work?>>

NOTE 2—The ICV (see <ref-3.X>) is often called the Message Authentication Code (MAC) in the security community, and is called Message Integrity Code (MIC) in IEEE 802.11. The term ICV is used in this standard to avoid confusion with the use of MAC in IEEE 802 Standards to mean Media Access Control, and with the specific use of MIC in 802.11.

NOTE 3—The crypto strength of the ICV is a function of its length, and is not tied to the data rate. It is therefore unnecessary to consider differences between various MAC technologies for ICV length selection [Discuss in section Z].

<<I worry that perhaps the SAID should be included within the scope of AAD. Is there a security exposure here? Since few SAIDs should be current at a receiver at any time, probably just 2 for point-to-point connectivity, exposure seems unlikely, but perhaps the question should be asked anyway. Omitting the SAID from ICV scope is quite different to omitting MAC specific fields, since those fields are never delivered to a service user and do not determine whether delivery takes place or not. They just get in the way of transparently bridging secured frames between different MACs.>>

9.4 Security Associations (SA)

<<haven't started to meld this clause in yet>>

The Security Association is the primary construct for establishing security. It defines the salient characteristics of secure communication between parties. A security association is a logical connection between two endpoints for the purposes of supporting secure communication between them.

Each security association carries with it parameters that control the operation of the Security Association. Such parameters would include:

1) Encryption algorithm and key length
2) Encryption algorithm key(s)
3) Message integrity algorithm and key length
4) Message integrity algorithm key(s)
5) key lifetime(s)
6) replay window size

Security Associations are created and destroyed through the use of a secure key agreement and security association management protocol, which negotiates these parameters between endpoints, such as IKE, Kerberos, or 802.1aa.

Security Associations can be uni-directional, or bi-directional. For a uni-directional association, the association applies only to traffic from A--->B, but not from B--->A. It is usually the case that security associations are negotiated in pairs, so that communication is always possible in both directions. In this case, a commit protocol is usually used to assure bi-directional communication.

A specific security association is usually identified by carrying a Security Association Identifier (SAID) in a special field with frame. Conceptually, this SAID is an index into a database (perhaps a quite small database) maintained at either end of a Security Association that contains the relevant security parameters. SAIDs have traditionally been large--on the order of 4 octets, in order to permit multiple SAs between two points in order to carry out various policies. However, in the case of MACsec, the number of necessary SAs between two points may be quite small, and the size of the SAID smaller.

9.5 Protocol support requirements

<<Just requirements for the system housing a SecY in this subclause, other requirements go in other clauses as called out in the introductory material to clause 9.1>>

<<Need a MAC address for use as an SOA, need to be able to store at least two sets of generation/validation parameter sets (one for each current SAID) and switch between them without missing a frame, no need to be able to do cipher suite to cipher suite seamless switching apart from to and from the Default Cipher Suite.>>

9.6 Cryptographic support requirements

<<Stuff about how to measure the qualitiative goodness of a cipher suite moved from here into Cipher Suite selection criteria. Just the absolute requirements here. Top of mind is the need for adequate AAD. Reference to speed criteria as included in the Interoperability requirements, operation over transparently bridged networks necessitating communication between peer SecYs at different speeds. The 10 Gig connects to the 1 Gig, the 1 Gig connects to the 100 Meg, the 100 Meg connects to the 10 Meg, so they all need the same manadatory default cipher suites. The exception is the current 802.11 wireless standards, since they connect to APs, not bridges, and are thus attached terminals not infrastructure components and are fundamentally unlikely to peer with other stations across a Provider Bridged Network. Therefore select for parallelizable algortihms and accept that wireless will be different - anyway no proposal to change 802.11i is wanted or intended. Additional cipher suites need to fit within the PDU expansion allowed.>>

9.7 Key Agreement support requirements

<<What MACsec depends on Key Agreement to achieve. Timeliness of operation etc. etc. Mention the policy and authorization implications/needs, or the requirement for binding to a secure tunnel that can supply these. Timing of coordinated changes. Deployability criteria not to be mentionned here as will be covered by the Key Agreement PAR/Project.>>

9.8 Cipher Suite Selection Criteria

<<Relocate after MACsec State Machines and Performance Parameter Management when both those clauses are flushed out. Some of this may be better in a permamnet Appendix, either Informative or Normative.>>

A cipher suite provides a set of cryptographic methods that provide confidentiality/privacy and integrity/ message authentication. Due to constraints imposed by implementation of these algorithms in hardware, we define a minimal default set of features mandatory to implement.

The important criteria for choosing cryptographic algorithms include:

1) crypto strength, provably secure << how is it measured?>>
2) peer review among cryptographers <<how measured?>>
3) exportability
4) interoperability <<??>>
5) ease of hardware implementation- memory, gate count
6) encumbrance status
7) speed - parallelizability

>>

A safe method of choosing cryptographic algorithms is to use cipher suites. A cipher suite fulfills the requirements for a well-defined minimal mandatory set of security features. A cipher suite specifies a set of algorithms for some or each of key exchange, data confidentiality, data origin (message) authentication, and replay protection. The interaction between the various algorithms that constitute the set of modes in a given cipher suite have been shown to interact well together and are well understood within the cryptography

community. Each cipher suite can be shown to be verifiably secure by security experts using formal methods.

<<Some of the combination mode stuff needs mentionning in 9.3 Cipher Suites>>

Some modes are called combination modes. These modes consist of cryptographic algorithms which offer multiple crypto functions within the same algorithm. For example, Encryption Authentication (EA) modes provide both confidentiality and integrity into one “combined” algorithm. The tag in a combined mode replaces the ICV. These modes can be included in cipher suites with other algorithms to provide additional functionality.

We will use only bona fide cipher suites, including combined modes. We will not make our own choice of cryptographic algorithms outside those which appear together in a given cipher suite, i.e., we will not mix and match crypto algorithms. This is to prevent security breaches similar to those that experience has shown are likely to occur when crypto algorithms are matched with each other.

Variable parameters (e.g., variable tag size in CCM, ref.) creates a vulnerability and therefore also leads to poor security. [say more, example to make clear what is meant].

<<do we want references?>>

<<See Appendix Z>>

One way to decide whether a crypto algorithm has various criteria is to follow guidance of FIPS/ NIST.<<how to judge whether crypto algorithms meet criteria? We should better understand FIPS process>>

The size of parameter fields in crypto algorithms: the cipher suite entry should specify the lengths of all the parameters for all algorithms used in the cipher suite. We will not dynamically vary parameter lengths. Different size parameters may be necessary for different media types. These can be handled by specifying different cipher suites offer different parameter sizes. See Appendix Z.

When parameter lengths are fixed, they algorithms are amenable to mathematical techniques, and can be provably secure.

Provider Bridges result in end to end connections (and SAs) between dissimilar technologies (e.g., 802.11 vs. 802.3). This difference in media technologies causes variations in cryptographic needs, such as the PN length, parallelizability, etc.

A and B will have the same default cipher suite, and either use that, or choose a vendor mode that is different than the default. Reason to negotiate even when on the same media is if can do better than the default.

9.9 Performance parameter management

Table 9-1 specifies default values and ranges for timer and transmission rate limiting performance parameters. Defaults are specified to avoid the need to set values prior to operation in most cases, and have been chosen for their wide applicability to maximize ease of operation. Ranges are specified to ensure that the protocol operates correctly, and provide guidance to implementors.

NOTE—Changes to Bridge Forward Delay do not affect reconfiguration times, unless the network includes Bridges that do not conform to this revision of this Standard. Changes to Bridge Max Age can have an effect, as it is possible for old information to persist in loops in the physical topology for a number of ‘hops’ equal to the value of Max Age in seconds, and thus exhaust the Transmit Hold Count in small loops.

Table 9-2—Performance parameters

Parameter	Recommended or Default value	Permitted Range	Compatibility Range

All times are in seconds. —[1] Not applicable, value is fixed.

9.10 MACsec state machines

The behavior of a SecY implementation in a Bridge is specified by a number of cooperating state machines. Figure 9-1 is not itself a state machine, but illustrates the machines, their interrelationships, the principal variables used to communicate between them, their local variables, and performance parameters.

One instance of each of the other state machines shall be implemented.

9.11 Notational conventions used in state diagrams

State diagrams are used to represent the operation of the protocol by a number of cooperating state machines each comprising a group of connected, mutually exclusive states. Only one state of each machine can be active at any given time.

Each state is represented in the state diagram as a rectangular box, divided into two parts by a horizontal line. The upper part contains the state identifier, written in upper case letters. The lower part contains any procedures that are executed on entry to the state.

All permissible transitions between states are represented by arrows, the arrowhead denoting the direction of the possible transition. Labels attached to arrows denote the condition(s) that must be met in order for the transition to take place. All conditions are expressions that evaluate to TRUE or FALSE; if a condition evaluates to TRUE, then the condition is met. The label UCT denotes an unconditional transition (i.e., UCT always evaluates to TRUE). A transition that is global in nature (i.e., a transition that occurs from any of the possible states if the condition attached to the arrow is met) is denoted by an open arrow; i.e., no specific state is identified as the origin of the transition. When the condition associated with a global transition is met, it supersedes all other exit conditions including UCT. The special global condition BEGIN supersedes all other global conditions, and once asserted remains asserted until all state blocks have executed to the point that variable assignments and other consequences of their execution remain unchanged.

On entry to a state, the procedures defined for the state (if any) are executed exactly once, in the order that they appear on the page. Each action is deemed to be atomic; i.e., execution of a procedure completes before the next sequential procedure starts to execute. No procedures execute outside of a state block. The procedures in only one state block execute at a time, even if the conditions for execution of state blocks in different state machines are satisfied, and all procedures in an executing state block complete execution before the transition to and execution of any other state block occurs, i.e. the execution of any state block appears to be atomic with respect to the execution of any other state block and the transition condition to that state from the previous state is TRUE when execution commences. The order of execution of state blocks in different state machines is undefined except as constrained by their transition conditions. A variable that is set to a particular value in a state block retains this value until a subsequent state block executes a procedure that modifies the value.

Figure 9-1—MACsec state machines - overview and interrelationships

On completion of all of the procedures within a state, all exit conditions for the state (including all conditions associated with global transitions) are evaluated continuously until one of the conditions is met. The label ELSE denotes a transition that occurs if none of the other conditions for transitions from the state are met (i.e., ELSE evaluates to TRUE if all other possible exit conditions from the state evaluate to FALSE). Where two or more exit conditions with the same level of precedence become TRUE simultaneously, the choice as to which exit condition causes the state transition to take place is arbitrary.

Where it is necessary to split a state machine description across more than one diagram, a transition between two states that appear on different diagrams is represented by an exit arrow drawn with dashed lines, plus a reference to the diagram that contains the destination state. Similarly, dashed arrows and a dashed state box are used on the destination diagram to show the transition to the destination state. In a state machine that has been split in this way, any global transitions that can cause entry to states defined in one of the diagrams are deemed to be potential exit conditions for all of the states of the state machine, regardless of which diagram the state boxes appear in.

Should a conflict exist between the interpretation of a state diagram and either the corresponding global transition tables or the textual description associated with the state machine, the state diagram takes precedence. The interpretation of the special symbols and operators used in the state diagrams is as defined in Table 9-3; these symbols and operators are derived from the notation of the “C++” programming language, ISO/IEC 14882. If a boolean variable is described in this clause as being set it has or is assigned the value TRUE, if reset or clear the value FALSE.

Table 9-3—State machine symbols

Symbol	Interpretation
( )	Used to force the precedence of operators in Boolean expressions and to delimit the argument(s) of actions within state boxes.
;	Used as a terminating delimiter for actions within state boxes. Where a state box contains multiple actions, the order of execution follows the normal English language conventions for reading text.
=	Assignment action. The value of the expression to the right of the operator is assigned to the variable to the left of the operator. Where this operator is used to define multiple assignments, e.g., a = b = X the action causes the value of the expression following the right-most assignment operator to be assigned to all of the variables that appear to the left of the right-most assignment operator.
!	Logical NOT operator.
&&	Logical AND operator.
||	Logical OR operator.
if...then...	Conditional action. If the Boolean expression following the if evaluates to TRUE, then the action following the then is executed.
!=	Inequality. Evaluates to TRUE if the expression to the left of the operator is not equal in value to the expression to the right.
==	Equality. Evaluates to TRUE if the expression to the left of the operator is equal in value to the expression to the right.
*	Arithmetic multiplication operator.
-	Arithmetic subtraction operator.

9.12 State machine timers

The timer variables declared in this clause, 9.12, are part of the specification of the operation of the SecY. The accompanying descriptions of their meaning and use are provided to aid in the comprehension of the protocol only, and are not part of the specification. A SecY implementation shall implement a single instance of each timer variable.

Each timer variable represents an integral number of seconds before timer expiry.

9.12.1 Timer 1

Timer 1.

9.13 State machine variables

The variables declared in this clause, 9.13, are part of the specification of the operation of the SecY. The accompanying descriptions of their use are provided to aid in the comprehension of the protocol only, and are not part of the specification.

9.13.1 BEGIN

A boolean controlled by the system initialization (9.11). If TRUE causes all state machines, including per Port state machines, to continuously execute their initial state.

9.13.2 Variable 1.

Variable 1.

9.13.3 portEnabled

A boolean. Set if the SecY can use the MAC Service provided by the Port's MAC entity to transmit and receive frames to and from the attached LAN, i.e. portEnabled is TRUE if and only if:

a) MAC_Operational (IEEE Std 802.1D Clause 6.4.2) is TRUE.

9.14 State machine conditions and parameters

The following variable evaluations are defined for notational convenience in the state machines.

9.14.1 Condition 1.

Condition 1.

9.15 State machine procedures

The following naming convention is used for the names of procedures that modify multiple variables (either multiple variables of a single Port or variables of multiple Ports):

a) ***set***: The procedure sets the value of the variables to TRUE.
b) ***clear:*** The procedure clears (resets) the value of the variables to FALSE.
c) ***updt:*** The procedure updates the variables in some other way.

The suffix "Tree" is used for procedures that can modify a variable in all Ports of the Bridge. For example, ***setSyncTree()*** is the name of a procedure that sets a variable TRUE for all Bridge Ports.

Where procedures are used to determine the value of a single variable, the procedure's returned value is explicitly assigned to the variable in the state machine concerned.

9.15.1 proc1()

Returns TRUE if....

9.16 XXX state machine

The XXX state machine shall implement the function specified by the state diagram in Figure 9-1, the definitions in 9.11, and the variable declarations and procedures specified in 9.12 through 9.15.

Figure 9-1—XXX state machine

9.17 SecY performance requirements

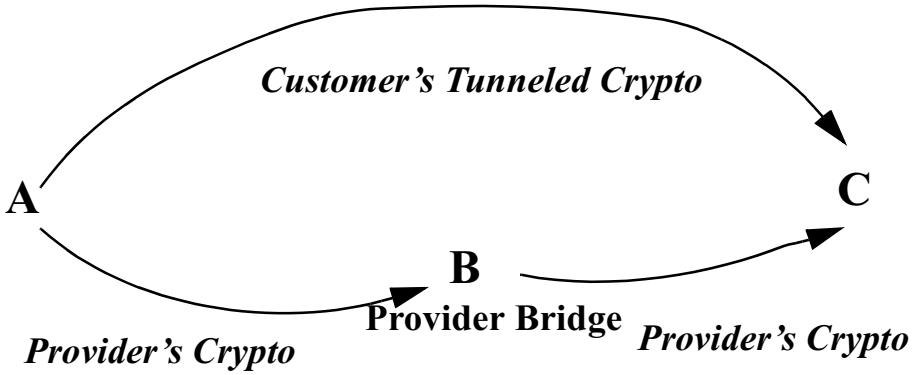
This clause (9.17) places requirements on the performance of the SecYs to ensure that MACsec operates correctly.

<<Some miscellaneous stuff>>

<<

<<The following (right up to the clause end) is material from (a) prior revision(s) of this clause (MACsec Protocol) that either needs to be placed somewhere in the revised organization of this clause or that indicates concerns that need to be addressed.>>

There are two level of cryptography taking place.



9.17.1 Additional Authenticated Data (AAD) (aka Additional Encrypted and Authenticated Data AEAD)

Certain fields need to be authenticated but not encrypted, specifically, in this context, the Source and Destination addresses. If a ciphertext is used that uses two different algorithms for encryption and authentication, there is no issue. The source address and destination addresses can be included in the ICV but not encrypted. However, if a combined mode is used, then the source and destination addresses will be encrypted in order to be authenticated, and they also need to be in the clear in the frame. Therefore additional fields may be prepended to the frame for fields which need to be authenticated but must also appear in the clear.

>>

10. Encoding of Secure MAC protocol data units

<<Material borrowed from 802.1D is scattered through this clause as a prompt to the editor and reviewers to supply analogous material for MAC Security, if appropriate.>>

This clause specifies the structure and encoding of the Secure MAC Protocol Data Units (SMPDUs) exchanged between MAC Security Entities (SecYs) for the MAC Security Protocol (MACsec) (Clause 9).

10.1 Structure

10.1.1 Transmission and representation of octets

All BPDUs shall contain an integral number of octets. The octets in a BPDU are numbered starting from 1 and increasing in the order they are put into a Data Link Service Data Unit (DLSDU). The bits in an octet are numbered from 1 to 8, where 1 is the low-order bit.

When consecutive bits within an octet are used to represent a binary number, the higher bit number has the most significant value. When consecutive octets are used to represent a binary number, the lower octet number has the most significant value. All Bridge Protocol Entities respect these bit and octet ordering conventions, thus allowing communications to take place.

10.1.2 Components

A Protocol Identifier is encoded in the initial octets of all SMPDUs. This standard specifies a single Protocol Identifier value for use in SMPDUs. All other Protocol Identifier values are reserved for future standards use. This standard places no further restriction on the structure, encoding, or use of SMPDUs with different values of the Protocol Identifier field, should these exist, by other standard protocols.

10.2 Encoding of parameter types

10.2.1 Encoding of protocol identifiers

A Protocol Identifier shall be encoded in two octets.

10.2.2 Encoding of protocol version identifiers

A Protocol Version Identifier shall be encoded in one octet. If two Protocol Version Identifiers are interpreted as unsigned binary numbers, the greater number identifies the more recently defined Protocol Version.

10.2.3 Encoding of BPDU types

The type of the BPDU shall be encoded as a single octet. The bit pattern contained in the octet merely serves to distinguish the type; no ordering relationship between BPDUs of different types is implied.

10.2.4 Encoding of flags

A flag shall be encoded as a bit in a single octet. A flag is set if the bit takes the value 1. A number of flags may be encoded in a single octet. Bits in the octet that do not correspond to flags defined for the BPDU's type are reset, i.e., shall take the value 0. No additional flags will be defined for a BPDU of given protocol version and type.

10.2.5 Encoding of Security Association Identifiers

.....

10.2.6 Encoding of Data Origin Identifiers

.....

10.3 SMPDU formats and parameters

10.3.1 Single hop point-to-point SMPDUs.

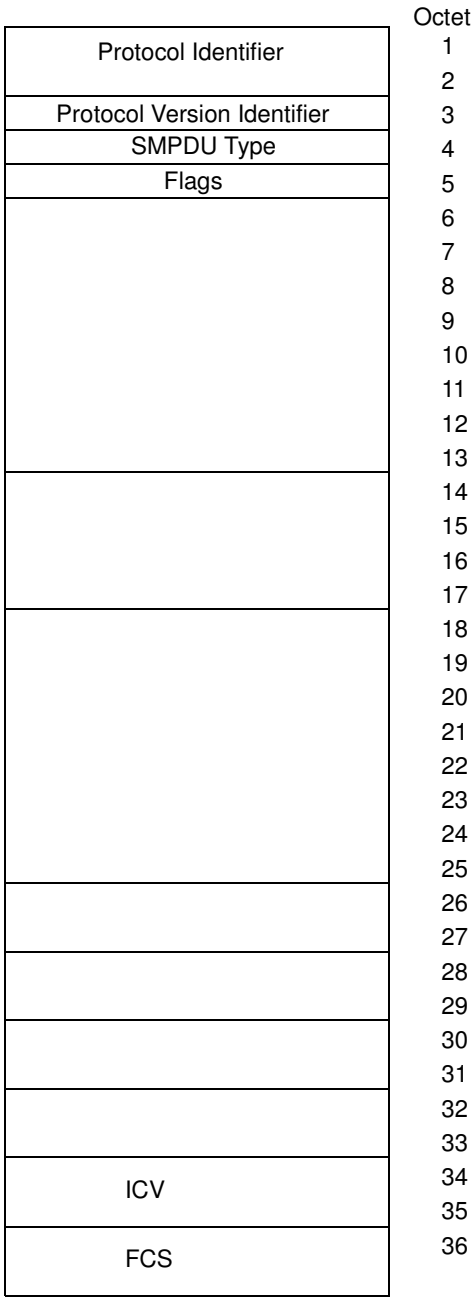


Figure 10-1—SMPDU parameters and format

11. Management of MAC Security Entities

This clause defines the set of managed objects, and their functionality, that allow administrative configuration and monitoring of MAC Security Entities.

This clause

a) Introduces the functions of management to assist in the identification of the requirements placed on MAC Security Entities for the support of management facilities
b) Establishes the correspondence between the state machines used to model the operation of a MAC Security Entity (SECy) () and its managed objects
c) Specifies the management operations supported by each managed object

11.1 Management functions

Management functions relate to the users' needs for facilities that support the planning, organization, supervision, control, protection, and security of communications resources, and account for their use. These facilities may be categorized as supporting the functional areas of Configuration, Fault, Performance, Security, and Accounting Management. Each of these is summarized in 11.1.1 through 11.1.5, together with the facilities commonly required for the management of communication resources, and the particular facilities provided in that functional area by MAC Security Entity Management.

11.1.1 Configuration Management

Configuration Management provides for the identification of communications resources, initialization, reset and close-down, the supply of operational parameters, and the establishment and discovery of the relationship between resources. The facilities provided by MAC Security Entity Management in this functional area are as follows:

a) Configuration of the operational parameters for the SECy (11.4.1.1 and 11.4.1.2)
b) Initialization of the state machines for the SECy ()

11.1.2 Fault Management

Fault Management provides for fault prevention, detection, diagnosis, and correction. The facilities provided by MAC Security Entity Management in this functional area are as follows:

a) Retrieval of SECy statistical information ()
b) Configuration of the operational parameters for the SECy (11.4.1.1 and 11.4.1.2)

11.1.3 Performance Management

Performance Management provides for evaluation of the behavior of communications resources and of the effectiveness of communication activities. The facilities provided by MAC Security Entity Management in this functional area are

a) Retrieval of statistical information ()
b) Configuration of the operational parameters (11.4.1.1 and 11.4.1.2)

11.1.4 Security Management

Security Management provides for the protection of resources. The facilities provided by MAC Security Entity Management in this functional area are as follows:

a) ???

11.1.5 Accounting Management

Accounting Management provides for the identification and distribution of costs and the setting of charges. The facilities provided by MAC Security Entity Management in this functional area is as follows:

a) Retrieval of accounting statistics (11.4.1.3)

11.2 Managed objects

Managed objects model the semantics of management operations. Operations upon a managed object supply information concerning, or facilitate control over, the Process or Entity associated with that managed object.

Management of MAC Security Entity is described in terms of the managed resources that are associated with individual Ports that support MAC Security. The managed resources of a SecY are those of the Processes and Entities established in... Specifically,

a) first sort of resource.....
b) .....

The management of these resources is described in terms of managed objects and operations defined below.

NOTE—The values specified in this clause, as inputs and outputs of management operations, are abstract information elements. Questions of format or encoding are a matter for particular protocols that convey or otherwise represent this information.

11.3 Data types

This sub clause specifies the semantics of operations independent of their encoding in management protocol. The data types of the parameters of operations are defined only as required for that specification.

The following data types are used:

a) Boolean
b) Enumerated, for a collection of named values
c) Unsigned, for all parameters specified as “the number of” some quantity
d) MAC Address
e) Time Interval, an Unsigned value representing a positive integral number of seconds, for all protocol time-out parameters
f) Counter, for all parameters specified as a “count” of some quantity (a counter increments and wraps with a modulus of 2 to the power of 64)

11.4 MAC Security Entity first sort of resource managed objects

The.....are described in.....

The objects that comprise this managed resource are as follows:

a) ....
b) ...

A MAC Security Entity that supports..... functionality shall support the management functionality defined by the... managed object. A MAC Security Entity that supports..... functionality may support the management functionality defined by the..... managed objects.

The means by which this management functionality is provided (e.g., the management protocol supported) shall be stated in the PICS associated with the implementation.

11.4.1 first resource first object

The.... managed object models the operations that modify, or enquire about, the configuration of the MAC Security Entity's resources. There is a single MAC Security Entity Configuration managed object for each MAC Security Entity.

The management operations that can be performed on the.... managed object are

a) Read.. Configuration (11.4.1.1)
b) Set.. Configuration (11.4.1.2)
c) ...(11.4.1.3)

11.4.1.1 Read... Configuration

11.4.1.1.1 Purpose

To solicit configuration information regarding the configuration of the MAC Security Entity.

11.4.1.1.2 Inputs

— **Identifier...**.

11.4.1.1.3 Outputs

a) **Identifier...** The identification number assigned to the MAC Security Entity**...**.
b) .....

11.4.1.2 Set... Configuration

11.4.1.2.1 Purpose

To configure the parameters that control the operation of the MAC Security Entity.

11.4.1.2.2 Inputs

Any parameters marked as (optional) may be omitted from the operation to allow selective modification of a subset of the configuration parameters. However, implementations shall support the ability to include all of the parameters identified below.

a) ....
b)
c)

11.4.1.2.3 Outputs

None.

11.4.1.3 .....

11.4.1.3.1 Purpose

...

11.4.1.3.2 Inputs

a) **...**

11.4.1.3.3 Outputs

None.

11.4.1.3.4 Effect

This operation...

12. Management protocol

12.1 Introduction

This clause defines a portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP-based internets. In particular, it defines objects for managing the operation of MAC Security, based on the specification contained in Clause 8 and Clause 1. This clause includes a MIB module that is SNMPv2 SMI compliant.

12.2 The SNMP Management Framework

The SNMP Management Framework presently consists of five major components:

a) An overall architecture, described in IETF RFC 2571.
b) Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIv1 and described in IETF RFC 1155, IETF RFC 1212, and IETF RFC 1215. The second version, called SMIv2, is described in IETF RFC 2578, IETF RFC 2579, and IETF RFC 2580.
c) Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in IETF RFC 1157. A second version of the SNMP message protocol, which is not an Internet standards track protocol, is called SNMPv2c and is described in IETF RFC 1901 and IETF RFC 1906. The third version of the message protocol is called SNMPv3 and is described in IETF RFC 1906, IETF RFC 2572 and IETF RFC 2574.
d) Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in IETF RFC 1157. A second set of protocol operations and associated PDU formats is described in IETF RFC 1905.
e) A set of fundamental applications described in IETF RFC 2573 and the view-based access control mechanism described in IETF RFC 2575.

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This clause specifies a MIB module that is compliant to the SMIv2. A MIB conforming to the SMIv1 can be produced through the appropriate translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine-readable information in SMIv2 will be converted into textual descriptions in SMIv1 during the translation process. However, this loss of machine-readable information is not considered to change the semantics of the MIB.

12.3 Security considerations

A number of management objects are defined in this MIB that have a MAX-ACCESS clause of read-write or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a nonsecure environment without proper protection can have a negative effect on network operations.

SNMPv1 by itself is not a secure environment. Even if the network is secure (for example, by using IPSec), there is no control as to who on the secure network is allowed to access (read/change/create/delete) the objects in this MIB.

It is recommended that the implementors consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model, IETF RFC 2574, and the View-based

Access Control Model, IETF RFC 2575, is recommended. It then becomes a user responsibility to ensure that the SNMP entity giving access to an instance of this MIB is properly configured to give access only to those principals (users) that have legitimate rights to access (change/create/delete) them, as appropriate.

12.4 Structure of the MIB

A single MIB module is defined in this clause. Objects in the MIB are arranged into groups. Each group is organized as a set of related objects. The overall structure and assignment of objects to their groups is shown in the following sub clauses.

12.4.1 Relationship to the managed objects defined in Clause 11

Table 12-1 contains cross-references between the objects defined in Clause 11 and the MIB objects defined in this clause.

Table 12-1—Managed object cross-reference table

Definition in Clause 11	MIB object(s)
11.4.1 first resource first object	**dot1aeMACsecy**
....	
....	
....	
....	
Initialize SECy	

Table 12-1—Managed object cross-reference table *(continued)*

Definition in Clause 11	MIB object(s)
Statistics	**dot1aeStatsTable**
Diagnostics	**dot1aeDiagTable**

12.4.2 The.....Group

This group of objects provides management functionality that is not specific to the operation of.....

12.4.3 The... Group

This group of objects provides.....

12.5 Relationship to other MIBs

It is assumed that a system implementing this MIB will also implement (at least) the "system" group defined in MIB-II defined in IETF RFC 1213 and the "interfaces" group defined in IETF RFC 2863.

12.5.1 Relationship to the Interfaces MIB

IETF RFC 2863, the Interface MIB Evolution, requires that any MIB that is an adjunct of the Interface MIB clarify specific areas within the Interface MIB. These areas were intentionally left vague in IETF RFC 2863 to avoid over constraining the MIB, thereby precluding management of certain media types.

Section 3.3 of IETF RFC 2863 enumerates several areas that a media-specific MIB must clarify. Each of these areas is addressed in a following subsection. The implementor is referred to IETF RFC 2863 in order to understand the general intent of these areas.

In IETF RFC 2863, the "interfaces" group is defined as being mandatory for all systems and contains information on an entity's interfaces, where each interface is thought of as being attached to a *subnetwork.* (Note that this term is not to be confused with *subnet,* which refers to an addressing partitioning scheme used in the Internet suite of protocols.) The term *segment* is sometimes used to refer to such a subnetwork.

Where Port numbers are used in this standard to identify Ports of a System, these numbers are equal to the ifIndex value for the interface for the corresponding Port.

12.6 Definitions for MAC Security MIB

In the MIB definition below, should any discrepancy between the DESCRIPTION text and the corresponding definition in Clause 11 occur, the definition in Clause 11 shall take precedence.

```
IEEE8021-SECY-MIB DEFINITIONS ::= BEGIN

-- ------------------------------------------------------- --
-- IEEE 802.1AE MIB
-- ------------------------------------------------------- --

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, Counter32, Counter64,
    Unsigned32, TimeTicks
        FROM SNMPv2-SMI
    MacAddress, TEXTUAL-CONVENTION, TruthValue
        FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP
        FROM SNMPv2-CONF
```

```
    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB
    InterfaceIndex
        FROM IF-MIB
    ;

ieee8021secyMIB MODULE-IDENTITY
    LAST-UPDATED "200307170000Z" -- 17th July 2003
    ORGANIZATION "IEEE 802.1 Working Group"
    CONTACT-INFO
        "http://grouper.ieee.org/groups/802/1/index.html"
    DESCRIPTION
        "The MAC Security Entity"
    REVISION "200307170000Z" -- 17th July 2003
    DESCRIPTION
        "Beginnings of a draft using the 802.1X MIB as a model"
::= { iso(1) std(0) iso8802(8802) ieee802dot1(1)
          ieee802dot1mibs(1) 1 }

paeMIBObjects OBJECT IDENTIFIER ::= { ieee8021paeMIB 1 }

-- ------------------------------------------------------------ --
-- Textual Conventions
-- ------------------------------------------------------------ --

PaeControlledDirections ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
        "The control mode values for the Authenticator PAE."
    SYNTAX      INTEGER {
                    both(0),
                    in(1)
                }

PaeControlledPortStatus ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "The status values of the Authenticator PAE controlled
        Port."
    SYNTAX      INTEGER {
                    authorized(1),
                    unauthorized(2)
                }

PaeControlledPortControl ::= TEXTUAL-CONVENTION
    STATUS       current
```

```
    DESCRIPTION
        "The control values of the Authenticator PAE controlled
        Port."
    SYNTAX      INTEGER {
                    forceUnauthorized(1),
                    auto(2),
                    forceAuthorized(3)
                }

-- ------------------------------------------------------------ --

-- ------------------------------------------------------------ --
-- groups in the PAE MIB
-- ------------------------------------------------------------ --

dot1xPaeSystem        OBJECT IDENTIFIER ::= { paeMIBObjects 1 }
dot1xPaeAuthenticator OBJECT IDENTIFIER ::= { paeMIBObjects 2 }
dot1xPaeSupplicant    OBJECT IDENTIFIER ::= { paeMIBObjects 3 }

-- ------------------------------------------------------------ --

-- ------------------------------------------------------------ --
-- The PAE System Group
-- ------------------------------------------------------------ --

dot1xPaeSystemAuthControl OBJECT-TYPE
    SYNTAX      INTEGER { enabled(1), disabled(2) }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The administrative enable/disable state for
         Port Access Control in a System."
    REFERENCE
        "<clause ref>, SystemAuthControl"
    ::= { dot1xPaeSystem 1 }

-- ------------------------------------------------------------ --
-- The PAE Port Table
-- ------------------------------------------------------------ --

dot1xPaePortTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xPaePortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of system level information for each port
```

```
        supported by the Port Access Entity.  An entry appears
        in this table for each port of this system."
    REFERENCE
        "<clause ref>"
    ::= { dot1xPaeSystem 2 }

dot1xPaePortEntry OBJECT-TYPE
    SYNTAX      Dot1xPaePortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The Port number, protocol version, and
        initialization control for a Port."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xPaePortTable 1 }

Dot1xPaePortEntry ::=
    SEQUENCE {
        dot1xPaePortNumber
            InterfaceIndex,
        dot1xPaePortProtocolVersion
            Unsigned32,
        dot1xPaePortCapabilities
            BITS,
        dot1xPaePortInitialize
            TruthValue,
        dot1xPaePortReauthenticate
            TruthValue
        }

dot1xPaePortNumber OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The Port number associated with this Port."
    REFERENCE
        "<clause ref>, Port number"
    ::= { dot1xPaePortEntry 1 }

dot1xPaePortProtocolVersion OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The protocol version associated with this Port."
```

```
    REFERENCE
        "<clause ref>, Protocol version"
    ::= { dot1xPaePortEntry 2 }

dot1xPaePortCapabilities OBJECT-TYPE
    SYNTAX      BITS {
                    dot1xPaePortAuthCapable(0),
                        -- Authenticator functions are supported
                    dot1xPaePortSuppCapable(1)
                        -- Supplicant functions are supported
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Indicates the PAE functionality that this Port
        supports and that may be managed through this MIB."
    REFERENCE
        "<clause ref>, PAE Capabilities"
    ::= { dot1xPaePortEntry 3 }

dot1xPaePortInitialize OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The initialization control for this Port. Setting this
        attribute TRUE causes the Port to be initialized.
        The attribute value reverts to FALSE once initialization
        has completed."
    REFERENCE
        "<clause ref>, Initialize Port"
    ::= { dot1xPaePortEntry 4 }

dot1xPaePortReauthenticate OBJECT-TYPE
  SYNTAX TruthValue
  MAX-ACCESS read-write
  STATUS current
  DESCRIPTION
    "The reauthentication control for this port. Setting
    this attribute TRUE causes the Authenticator PAE state
    machine for the Port to reauthenticate the Supplicant.
    Setting this attribute FALSE has no effect.
    This attribute always returns FALSE when it is read."
  REFERENCE
    "<clause ref> Reauthenticate"
::= { dot1xPaePortEntry 5 }
```

```
-- ------------------------------------------------------- --
-- The PAE Authenticator Group
-- ------------------------------------------------------- --

-- ------------------------------------------------------- --
-- The Authenticator Configuration Table
-- ------------------------------------------------------- --

dot1xAuthConfigTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xAuthConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table that contains the configuration objects for the
        Authenticator PAE associated with each port.
        An entry appears in this table for each port that may
        authenticate access to itself."
    REFERENCE
        "<clause ref> Authenticator Configuration"
    ::= { dot1xPaeAuthenticator 1 }

dot1xAuthConfigEntry OBJECT-TYPE
    SYNTAX      Dot1xAuthConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The configuration information for an Authenticator
        PAE."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xAuthConfigTable 1 }

Dot1xAuthConfigEntry ::=
    SEQUENCE {
        dot1xAuthPaeState
            INTEGER,
        dot1xAuthBackendAuthState
            INTEGER,
        dot1xAuthAdminControlledDirections
            PaeControlledDirections,
        dot1xAuthOperControlledDirections
            PaeControlledDirections,
        dot1xAuthAuthControlledPortStatus
            PaeControlledPortStatus,
        dot1xAuthAuthControlledPortControl
            PaeControlledPortControl,
```

```
        dot1xAuthQuietPeriod
            Unsigned32,
        dot1xAuthTxPeriod
            Unsigned32,
        dot1xAuthSuppTimeout (Deprecated)
            Unsigned32,
        dot1xAuthServerTimeout
            Unsigned32,
        dot1xAuthMaxReq (Deprecated, value is ignored)
            Unsigned32,
        dot1xAuthReAuthPeriod
            Unsigned32,
        dot1xAuthReAuthEnabled
            TruthValue,
        dot1xAuthKeyTxEnabled
            TruthValue,
        dot1xAuthInitEapCode
            INTEGER,
        dot1xAuthInitEapData
            OCTET STRING
        }

dot1xAuthPaeState OBJECT-TYPE
    SYNTAX      INTEGER {
                    initialize(1),
                    disconnected(2),
                    connecting(3),
                    authenticating(4),
                    authenticated(5),
                    aborting(6),
                    held(7),
                    forceAuth(8),
                    forceUnauth(9),
                    restart(10)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current value of the Authenticator PAE state
        machine."
    REFERENCE
        "<clause ref>, Authenticator PAE state"
    ::= { dot1xAuthConfigEntry 1 }

dot1xAuthBackendAuthState OBJECT-TYPE
    SYNTAX      INTEGER {
```

```
                    request(1),
                    response(2),
                    success(3),
                    fail(4),
                    timeout(5),
                    idle(6),
                    initialize(7),
                    ignore(8)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current state of the Backend Authentication
        state machine."
    REFERENCE
        "<clause ref>, Backend Authentication state"
    ::= { dot1xAuthConfigEntry 2 }

dot1xAuthAdminControlledDirections OBJECT-TYPE
    SYNTAX      PaeControlledDirections
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The current value of the administrative controlled
        directions parameter for the Port."
    REFERENCE
        "<clause ref>, Admin Control Mode"
    ::= { dot1xAuthConfigEntry 3 }

dot1xAuthOperControlledDirections OBJECT-TYPE
    SYNTAX      PaeControlledDirections
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current value of the operational controlled
        directions parameter for the Port."
    REFERENCE
        "<clause ref>, Oper Control Mode"
    ::= { dot1xAuthConfigEntry 4 }

dot1xAuthAuthControlledPortStatus OBJECT-TYPE
    SYNTAX      PaeControlledPortStatus
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current value of the controlled Port
```

```
        status parameter for the Port."
    REFERENCE
        "<clause ref>, AuthControlledPortStatus"
    ::= { dot1xAuthConfigEntry 5 }

dot1xAuthAuthControlledPortControl OBJECT-TYPE
    SYNTAX      PaeControlledPortControl
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The current value of the controlled Port
        control parameter for the Port."
    REFERENCE
        "<clause ref>, AuthControlledPortControl"
    ::= { dot1xAuthConfigEntry 6 }

dot1xAuthQuietPeriod OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the quietPeriod constant
        currently in use by the Authenticator PAE state
        machine."
    REFERENCE
        "<clause ref>, quietPeriod"
    DEFVAL { 60 }
    ::= { dot1xAuthConfigEntry 7 }

dot1xAuthSuppTimeout OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
        "Use of this valueis deprecated."
    REFERENCE
        "No reference"
    DEFVAL { 30 }
    ::= { dot1xAuthConfigEntry 9 }

dot1xAuthServerTimeout OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the serverTimeout constant
```

```
        currently in use by the Backend Authentication state
        machine."
    REFERENCE
        "<clause ref>, serverTimeout"
    DEFVAL { 30 }
    ::= { dot1xAuthConfigEntry 10 }

dot1xAuthMaxReq OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
        "This value is deprecated and is ignored."
    REFERENCE
        "There is no reference, the use of this value is deprecated."
    DEFVAL { 2 }
    ::= { dot1xAuthConfigEntry 11 }

dot1xAuthReAuthPeriod OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the reAuthPeriod constant
        currently in use by the Reauthentication Timer state
        machine."
    REFERENCE
        "<clause ref>, reAuthPeriod"
    DEFVAL { 3600 }
    ::= { dot1xAuthConfigEntry 12 }

dot1xAuthReAuthEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The enable/disable control used by the Reauthentication
        Timer state machine (<clause ref>)."
    REFERENCE
        "<clause ref>, reAuthEnabled"
    DEFVAL { false }
    ::= { dot1xAuthConfigEntry 13 }

dot1xAuthKeyTxEnabled OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
```

```
    STATUS      current
    DESCRIPTION
        "The value of the keyTransmissionEnabled constant
        currently in use by the Authenticator PAE state
        machine."
    REFERENCE
        "<clause ref>, keyTransmissionEnabled"
    ::= { dot1xAuthConfigEntry 14 }

<<Editor's Note: I think the following object is no longer needed.>>

dot1xAuthInitEapCode OBJECT-TYPE
    SYNTAX      INTEGER {
                    request(1),
                    response(2),
                    success(3),
                    failure(4)
                }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value of the EAP Code field to be used by the
        txInitialMsg() function of the Authenticator PAE state
        machine."
    REFERENCE
        "<clause ref>, initialEAPMsg; RFC 2284, section 2.2"
    DEFVAL { request }
    ::= { dot1xAuthConfigEntry 15 }

<<Editor's Note: I think the following object is no longer needed.>>

dot1xAuthInitEapData OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..1477))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value used to derive the EAP Data field by the
        txInitialMsg() function of the Authenticator PAE state
        machine.

        The other fields of the initial EAP message generated
        by txInitialMsg() are:

            EAP Code is taken from dot1xAuthInitEapCode

            EAP Identifier is the currentId passed into
```

```
            txInitialMsg().

            EAP Length is derived from the dot1xAuthInitEapData
            value.

        The derived EAP Data field itself depends upon the
        first byte of this object, which corresponds to the EAP
        Type value.  In the simplest case, such as for an EAP
        Identity (EAP Type 1) this may be a direct copy of the
        value of this object and the EAP Length is derived from
        the length of this object, plus 4 octes for the EAP
        Code, Identifier and Length values.  In more complex
        cases, such as an EAP MD5-Challenge (EAP Type 4), this
        may require one-time generated values to be inserted at
        the appropriate place in the data by the txInitialMsg()
        function and the EAP Length must be calculated based on
        the generated data.

        The size of this object is defined to support the
        maximum size that will fit in an EAPOL message.  This
        exceeds the size that may be carried in an SNMP message
        without requiring IP fragmentation on an Ethernet
        network.  It is expected that actual values used in
        this object will be much smaller than the maximum
        allowed.

        The default value for this object is an octet string
        containing a single octet with a value of 01 (hex).
        This corresponds to the EAP Type 'Identity', without
        the optional Type-Data field."
    REFERENCE
        "<clause ref>, initialEAPMsg; RFC 2284, section 3"
    DEFVAL { '01'H }                    -- EAP Type 'Identity'
    ::= { dot1xAuthConfigEntry 16 }

-- ---------------------------------------------------------- --
-- The Authenticator Statistics Table
-- ---------------------------------------------------------- --

dot1xAuthStatsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xAuthStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table that contains the statistics objects for the
         Authenticator PAE associated with each Port.
```

```
         An entry appears in this table for each port that may
         authenticate access to itself."
    REFERENCE
        "9.4.1 Authenticator Statistics"
    ::= { dot1xPaeAuthenticator 2 }

dot1xAuthStatsEntry OBJECT-TYPE
    SYNTAX      Dot1xAuthStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The statistics information for an Authenticator PAE."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xAuthStatsTable 1 }

Dot1xAuthStatsEntry ::=
    SEQUENCE {
        dot1xAuthEapolFramesRx
            Counter32,
        dot1xAuthEapolFramesTx
            Counter32,
        dot1xAuthEapolStartFramesRx
            Counter32,
        dot1xAuthEapolLogoffFramesRx
            Counter32,
        dot1xAuthEapolRespFramesRx
            Counter32,
        dot1xAuthEapolReqIdFramesTx
            Counter32,
        dot1xAuthEapolReqFramesTx
            Counter32,
        dot1xAuthInvalidEapolFramesRx
            Counter32,
        dot1xAuthEapLengthErrorFramesRx
            Counter32,
        dot1xAuthLastEapolFrameVersion
            Unsigned32,
        dot1xAuthLastEapolFrameSource
            MacAddress
        }

dot1xAuthEapolFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
```

```
        "The number of valid EAPOL frames of any type
        that have been received by this Authenticator."
    REFERENCE
        "<clause ref>, EAPOL frames received"
    ::= { dot1xAuthStatsEntry 1 }

dot1xAuthEapolFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames of any type
        that have been transmitted by this Authenticator."
    REFERENCE
        "<clause ref>, EAPOL frames transmitted"
    ::= { dot1xAuthStatsEntry 2 }

dot1xAuthEapolStartFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL Start frames that have
        been received by this Authenticator."
    REFERENCE
        "<clause ref>, EAPOL Start frames received"
    ::= { dot1xAuthStatsEntry 3 }

dot1xAuthEapolLogoffFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL Logoff frames that have
        been received by this Authenticator."
    REFERENCE
        "<clause ref>, EAPOL Logoff frames received"
    ::= { dot1xAuthStatsEntry 4 }

dot1xAuthEapolRespFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of valid EAPOL frames
        of type EAP-Packet that have been
```

```
        received by this Authenticator."
    REFERENCE
        "<clause ref>, EAPOL Response frames received"
    ::= { dot1xAuthStatsEntry 6 }

dot1xAuthEapolReqIdFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of initial EAPOL frames
        of type EAP-Packet that have been
        transmitted by this Authenticator
        prior to the first response from the
        Supplicant. This counts all txReqs carried
        out from the REQUEST state after transitioning
        from the IDLE state but before transitioning
        to the RESPONSE state."
    REFERENCE
        "<clause ref>, EAPOL Initial Request frames transmitted"
    ::= { dot1xAuthStatsEntry 7 }

dot1xAuthEapolReqFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames of type EAP-Packet
        (other than initial frames) that have been
        transmitted by this Authenticator. This counts
        all txReqs carried out from the REQUEST state
        after transitioning in from any state other
        than the IDLE state."
    REFERENCE
        "<clause ref>, EAPOL Request frames transmitted"
    ::= { dot1xAuthStatsEntry 8 }

dot1xAuthInvalidEapolFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames that have been
        received by this Authenticator in which the
        frame type is not recognized."
    REFERENCE
```

```
        "<clause ref>, Invalid EAPOL frames received"
    ::= { dot1xAuthStatsEntry 9 }

dot1xAuthEapLengthErrorFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames that have been received
        by this Authenticator in which the Packet Body
        Length field is invalid."
    REFERENCE
        "<clause ref>, EAPOL length error frames received"
    ::= { dot1xAuthStatsEntry 10 }

dot1xAuthLastEapolFrameVersion OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The protocol version number carried in the
        most recently received EAPOL frame."
    REFERENCE
        "<clause ref>, Last EAPOL frame version"
    ::= { dot1xAuthStatsEntry 11 }

dot1xAuthLastEapolFrameSource OBJECT-TYPE
    SYNTAX      MacAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The source MAC address carried in the
        most recently received EAPOL frame."
    REFERENCE
        "<clause ref>, Last EAPOL frame source"
    ::= { dot1xAuthStatsEntry 12 }

-- ---------------------------------------------------------- --
-- The Authenticator Diagnostics Table
-- ---------------------------------------------------------- --

dot1xAuthDiagTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xAuthDiagEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
```

```
        "A table that contains the diagnostics objects for the
         Authenticator PAE associated with each Port.
         An entry appears in this table for each port that may
         authenticate access to itself."
    REFERENCE
        "<clause ref> Authenticator Diagnostics"
    ::= { dot1xPaeAuthenticator 3 }

dot1xAuthDiagEntry OBJECT-TYPE
    SYNTAX      Dot1xAuthDiagEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The diagnostics information for an Authenticator PAE."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xAuthDiagTable 1 }

Dot1xAuthDiagEntry ::=
    SEQUENCE {
        dot1xAuthEntersConnecting
            Counter32,
        dot1xAuthEapLogoffsWhileConnecting
            Counter32,
        dot1xAuthEntersAuthenticating
            Counter32,
        dot1xAuthAuthSuccessWhileAuthenticating
            Counter32,
        dot1xAuthAuthTimeoutsWhileAuthenticating
            Counter32,
        dot1xAuthAuthFailWhileAuthenticating
            Counter32,
        dot1xAuthAuthReauthsWhileAuthenticating
            Counter32,
        dot1xAuthAuthEapStartsWhileAuthenticating
            Counter32,
        dot1xAuthAuthEapLogoffWhileAuthenticating
            Counter32,
        dot1xAuthAuthReauthsWhileAuthenticated (deprecated)
            Counter32,
        dot1xAuthAuthEapStartsWhileAuthenticated
            Counter32,
        dot1xAuthAuthEapLogoffWhileAuthenticated
            Counter32,
        dot1xAuthBackendResponses
            Counter32,
        dot1xAuthBackendAccessChallenges
```

```
            Counter32,
        dot1xAuthBackendOtherRequestsToSupplicant
            Counter32,
        dot1xAuthBackendAuthSuccesses
            Counter32,
        dot1xAuthBackendAuthFails
            Counter32
        }

dot1xAuthEntersConnecting OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions to the CONNECTING state from any other
        state."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 1 }

dot1xAuthEapLogoffsWhileConnecting OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from CONNECTING to DISCONNECTED as a result
        of receiving an EAPOL-Logoff message."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 2 }

dot1xAuthEntersAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from CONNECTING to AUTHENTICATING, as a
        result of an EAP-Response/Identity message being
        received from the Supplicant."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 3 }
```

```
dot1xAuthAuthSuccessWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATING to AUTHENTICATED, as a
        result of the Backend Authentication state machine
        indicating successful authentication of the Supplicant
        (authSuccess = TRUE)."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 4 }

dot1xAuthAuthTimeoutsWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATING to ABORTING, as a result
        of the Backend Authentication state machine indicating
        authentication timeout (authTimeout = TRUE)."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 5 }

dot1xAuthAuthFailWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATING to HELD, as a result
        of the Backend Authentication state machine indicating
        authentication failure (authFail = TRUE)."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 6 }

dot1xAuthAuthReauthsWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      deprecated
    DESCRIPTION
        "Use of this value is deprecated."
```

```
    REFERENCE
        "None"
    ::= { dot1xAuthDiagEntry 7 }

dot1xAuthAuthEapStartsWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATING to ABORTING, as a result
        of an EAPOL-Start message being received
        from the Supplicant."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 8 }

dot1xAuthAuthEapLogoffWhileAuthenticating OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATING to ABORTING, as a result
        of an EAPOL-Logoff message being received
        from the Supplicant."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 9 }

dot1xAuthAuthReauthsWhileAuthenticated OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATED to CONNECTING, as a
        result of a reauthentication request
        (reAuthenticate = TRUE)."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 10 }

dot1xAuthAuthEapStartsWhileAuthenticated OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATED to CONNECTING, as a
        result of an EAPOL-Start message being received from the
        Supplicant."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 11 }

dot1xAuthAuthEapLogoffWhileAuthenticated OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        transitions from AUTHENTICATED to DISCONNECTED, as a
        result of an EAPOL-Logoff message being received from
        the Supplicant."
    REFERENCE
        "<clause ref>, <clause ref>"
    ::= { dot1xAuthDiagEntry 12 }

dot1xAuthBackendResponses OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine sends
        a supplicant's first response packet to the EAP layer
        (i.e., executes sendRespToServer on entry to the
        RESPONSE state). Indicates that the Authenticator
        attempted communication with the Authentication Server
        via the EAP layer."
    REFERENCE
        "<clause ref>"
    ::= { dot1xAuthDiagEntry 13 }

dot1xAuthBackendAccessChallenges OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        receives the first request from the EAP layer following
        the first response from the supplicant (i.e., eapReq
```

```
        becomes TRUE, causing exit from the RESPONSE state).
        Indicates that the Authentication Server has
        communication with the Authenticator via the EAP layer."
    REFERENCE
        "<clause ref>"
    ::= { dot1xAuthDiagEntry 14 }

dot1xAuthBackendOtherRequestsToSupplicant OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        sends an EAP-Request packet following the first
        to the Supplicant (i.e., executes txReq on entry to the
        REQUEST state). Indicates that the Authentication
        Server chose an EAP-method."
    REFERENCE
        "<clause ref>"
    ::= { dot1xAuthDiagEntry 15 }

dot1xAuthBackendAuthSuccesses OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        receives a success indication from the EAP layer
        (i.e., aSuccess becomes TRUE, causing a
        transition from RESPONSE to SUCCESS). Indicates that
        the Supplicant has successfully authenticated to
        the Authentication Server."
    REFERENCE
        "<clause ref>"
    ::= { dot1xAuthDiagEntry 17 }

dot1xAuthBackendAuthFails OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Counts the number of times that the state machine
        receives a failure message from the EAP layer
        (i.e., aFail becomes TRUE, causing a transition
        from RESPONSE to FAIL). Indicates that the Supplicant
        has not authenticated to the Authentication Server."
```

```
    REFERENCE
        "<clause ref>"
    ::= { dot1xAuthDiagEntry 18 }

-- ---------------------------------------------------------- --
-- The Authenticator Session Statistics Table
-- ---------------------------------------------------------- --

dot1xAuthSessionStatsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xAuthSessionStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table that contains the session statistics objects
        for the Authenticator PAE associated with each Port.
        An entry appears in this table for each port that may
        authenticate access to itself."
    REFERENCE
        "<clause ref>"
    ::= { dot1xPaeAuthenticator 4 }

dot1xAuthSessionStatsEntry OBJECT-TYPE
    SYNTAX      Dot1xAuthSessionStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The session statistics information for an Authenticator
        PAE.  This shows the current values being collected for
        each session that is still in progress, or the final
        values for the last valid session on each port where
        there is no session currently active."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xAuthSessionStatsTable 1 }

Dot1xAuthSessionStatsEntry ::=
    SEQUENCE {
        dot1xAuthSessionOctetsRx
            Counter64,
        dot1xAuthSessionOctetsTx
            Counter64,
        dot1xAuthSessionFramesRx
            Counter32,
        dot1xAuthSessionFramesTx
            Counter32,
        dot1xAuthSessionId
            SnmpAdminString,
```

```
        dot1xAuthSessionAuthenticMethod
            INTEGER,
        dot1xAuthSessionTime
            TimeTicks,
        dot1xAuthSessionTerminateCause
            INTEGER,
        dot1xAuthSessionUserName
            SnmpAdminString
    }

dot1xAuthSessionOctetsRx OBJECT-TYPE
    SYNTAX      Counter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of octets received in user data
        frames on this Port during the session."
    REFERENCE
        "<clause ref>, Session Octets Received"
    ::= { dot1xAuthSessionStatsEntry 1 }

dot1xAuthSessionOctetsTx OBJECT-TYPE
    SYNTAX      Counter64
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of octets transmitted in user data
        frames on this Port during the session."
    REFERENCE
        "<clause ref>, Session Octets Transmitted"
    ::= { dot1xAuthSessionStatsEntry 2 }

dot1xAuthSessionFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of user data frames received
        on this Port during the session."
    REFERENCE
        "<clause ref>, Session Frames Received"
    ::= { dot1xAuthSessionStatsEntry 3 }

dot1xAuthSessionFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
        "The number of user data frames transmitted
        on this Port during the session."
    REFERENCE
        "<clause ref>, Session Frames Transmitted"
    ::= { dot1xAuthSessionStatsEntry 4 }

dot1xAuthSessionId OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "A unique identifier for the session, in the
        form of a printable ASCII string of at least
        three characters."
    REFERENCE
        "<clause ref>, Session Identifier"
    ::= { dot1xAuthSessionStatsEntry 5 }

dot1xAuthSessionAuthenticMethod OBJECT-TYPE
    SYNTAX      INTEGER {
                    remoteAuthServer(1),
                    localAuthServer(2)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The authentication method used to establish the
        session."
    REFERENCE
        "<clause ref>, Session Authentication Method"
    ::= { dot1xAuthSessionStatsEntry 6 }

dot1xAuthSessionTime OBJECT-TYPE
    SYNTAX      TimeTicks
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The duration of the session in seconds."
    REFERENCE
        "<clause ref>, Session Time"
    ::= { dot1xAuthSessionStatsEntry 7 }

dot1xAuthSessionTerminateCause OBJECT-TYPE
    SYNTAX      INTEGER {
```

```
                    supplicantLogoff(1),
                    portFailure(2),
                    supplicantRestart(3),
                    reauthFailed(4),
                    authControlForceUnauth(5),
                    portReInit(6),
                    portAdminDisabled(7),
                    notTerminatedYet(999)
                    }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The reason for the session termination."
    REFERENCE
        "<clause ref>, Session Terminate Cause"
    ::= { dot1xAuthSessionStatsEntry 8 }

dot1xAuthSessionUserName OBJECT-TYPE
    SYNTAX     SnmpAdminString
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The User-Name representing the identity of the
        Supplicant PAE."
    REFERENCE
        "<clause ref>, Session User Name"
    ::= { dot1xAuthSessionStatsEntry 9 }

-- ------------------------------------------------------- --
-- The PAE Supplicant Group
-- ------------------------------------------------------- --

-- ------------------------------------------------------- --
-- The Supplicant Configuration Table
-- ------------------------------------------------------- --

dot1xSuppConfigTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xSuppConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table that contains the configuration objects for the
        Supplicant PAE associated with each port.
        An entry appears in this table for each port that may
        authenticate itself when challenged by a remote system."
    REFERENCE
```

```
        "<clause ref>"
    ::= { dot1xPaeSupplicant 1 }

dot1xSuppConfigEntry OBJECT-TYPE
    SYNTAX      Dot1xSuppConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The configuration information for a Supplicant PAE."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xSuppConfigTable 1 }

Dot1xSuppConfigEntry ::=
    SEQUENCE {
        dot1xSuppPaeState
            INTEGER,
        dot1xSuppHeldPeriod
            Unsigned32,
        dot1xSuppAuthPeriod
            Unsigned32,
        dot1xSuppStartPeriod
            Unsigned32,
        dot1xSuppMaxStart
            Unsigned32,
        dot1xSuppBackendPaeState
            Unsigned32
        dot1xSuppSuppControlledPortStatus
            Unsigned32
        }

dot1xSuppPaeState OBJECT-TYPE
    SYNTAX      INTEGER {
                    disconnected(1),
                    logoff(2),
                    connecting(3),
                    authenticating(4),
                    authenticated(5),
                    unused(6),
                    held(7),
                    restart(8)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current state of the Supplicant PAE state
        machine (8.5.8)."
```

```
    REFERENCE
        "<clause ref>, Supplicant PAE State"
    ::= { dot1xSuppConfigEntry 1 }

dot1xSuppHeldPeriod OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the heldPeriod
        constant currently in use by the Supplicant
        PAE state machine (8.5.8.1.2)."
    REFERENCE
        "<clause ref>, heldPeriod"
    DEFVAL { 60 }
    ::= { dot1xSuppConfigEntry 2 }

dot1xSuppAuthPeriod OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the serverTimeout
        constant currently in use by the Supplicant
        PAE state machine (8.5.8.1.2)."
    REFERENCE
        "<clause ref>, authPeriod"
    DEFVAL { 30 }
    ::= { dot1xSuppConfigEntry 3 }

dot1xSuppStartPeriod OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "The value, in seconds, of the suppTimeout
        constant currently in use by the Supplicant
        PAE state machine (8.5.8.1.2)."
    REFERENCE
        "<clause ref>, startPeriod"
    DEFVAL { 30 }
    ::= { dot1xSuppConfigEntry 4 }

dot1xSuppMaxStart OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-write
```

```
    STATUS      current
    DESCRIPTION
        "The value of the maxStart constant currently in use by
        the Supplicant PAE state machine (8.5.8.1.2)."
    REFERENCE
        "<clause ref>, maxStart"
    DEFVAL { 3}
    ::= { dot1xSuppConfigEntry 5 }

dot1xSuppBackendPaeState OBJECT-TYPE
    SYNTAX INTEGER {
    initialize(1),
    idle(2),
    request(3),
    receive(4),
    response(5),
    fail(6),
    timeout(7),
    success(8)
    }
    MAX-ACCESS read-only
    STATUS current
    DESCRIPTION
        "The current state of the Backend Supplicant
         PAE statemachine (8.5.12)."
    REFERENCE
        "<clause ref>, Backend Supplicant PAE State"
    ::= { dot1xSuppConfigEntry 6 }

dot1xSuppSuppControlledPortStatus OBJECT-TYPE
    SYNTAX      PaeControlledPortStatus
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The current value of the controlled Port
        status parameter for the Port."
    REFERENCE
        "<clause ref>, SuppControlledPortStatus"
    ::= { dot1xSuppConfigEntry 7 }

-- ---------------------------------------------------------- --
-- The Supplicant Statistics Table
-- ---------------------------------------------------------- --

dot1xSuppStatsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Dot1xSuppStatsEntry
```

```
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table that contains the statistics objects for the
        Supplicant PAE associated with each port.
        An entry appears in this table for each port that may
        authenticate itself when challenged by a remote system."
    REFERENCE
        "<clause ref>"
    ::= { dot1xPaeSupplicant 2 }

dot1xSuppStatsEntry OBJECT-TYPE
    SYNTAX      Dot1xSuppStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "The statistics information for a Supplicant PAE."
    INDEX { dot1xPaePortNumber }
    ::= { dot1xSuppStatsTable 1 }

Dot1xSuppStatsEntry ::=
    SEQUENCE {
        dot1xSuppEapolFramesRx
            Counter32,
        dot1xSuppEapolFramesTx
            Counter32,
        dot1xSuppEapolStartFramesTx
            Counter32,
        dot1xSuppEapolLogoffFramesTx
            Counter32,
        dot1xSuppEapolRespIdFramesTx
            Counter32,
        dot1xSuppEapolRespFramesTx
            Counter32,
        dot1xSuppEapolReqIdFramesRx
            Counter32,
        dot1xSuppEapolReqFramesRx
            Counter32,
        dot1xSuppInvalidEapolFramesRx
            Counter32,
        dot1xSuppEapLengthErrorFramesRx
            Counter32,
        dot1xSuppLastEapolFrameVersion
            Unsigned32,
        dot1xSuppLastEapolFrameSource
            MacAddress
```

```
        }

dot1xSuppEapolFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames of any type
        that have been received by this Supplicant."
    REFERENCE
        "<clause ref>, EAPOL frames received"
    ::= { dot1xSuppStatsEntry 1 }

dot1xSuppEapolFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames of any type
        that have been transmitted by this Supplicant."
    REFERENCE
        "<clause ref>, EAPOL frames transmitted"
    ::= { dot1xSuppStatsEntry 2 }

dot1xSuppEapolStartFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL Start frames
        that have been transmitted by this Supplicant."
    REFERENCE
        "<clause ref>, EAPOL Start frames transmitted"
    ::= { dot1xSuppStatsEntry 3 }

dot1xSuppEapolLogoffFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL Logoff frames
        that have been transmitted by this Supplicant."
    REFERENCE
        "<clause ref>, EAPOL Logoff frames transmitted"
    ::= { dot1xSuppStatsEntry 4 }
```

```
dot1xSuppEapolRespFramesTx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of valid EAPOL frames of type EAP-Packet
        that have been transmitted by this Supplicant."
    REFERENCE
        "<clause ref>, EAP Resp frames transmitted"
    ::= { dot1xSuppStatsEntry 6 }

dot1xSuppEapolReqIdFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of initial EAPOL frames of type EAP-Packet
        that have been received by this Supplicant. This
        counts all txSuppRsp() carried out from the
        RESPONSE state after transitioning from the IDLE
        state but before transitioning to the RECEIVE state."
    REFERENCE
        "<clause ref>, EAP Initial Request frames received"
    ::= { dot1xSuppStatsEntry 7 }

dot1xSuppEapolReqFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames of type EAP-Packet
        (other than initial frames) that have been
        received by this Supplicant. This counts all
        txSuppRsp() carried out from the RESPONSE state
        after having passed once through the RECEIVE
        state and having not passed through the IDLE state."
    REFERENCE
        "<clause ref>, EAP Req frames received"
    ::= { dot1xSuppStatsEntry 8 }

dot1xSuppInvalidEapolFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames that have been
```

```
        received by this Supplicant in which the
        frame type is not recognized."
    REFERENCE
        "<clause ref>, Invalid EAPOL frames received"
    ::= { dot1xSuppStatsEntry 9 }

dot1xSuppEapLengthErrorFramesRx OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The number of EAPOL frames that have been
        received by this Supplicant in which the Packet
        Body Length field (7.5.5) is invalid."
    REFERENCE
        "<clause ref>, EAPOL length error frames received"
    ::= { dot1xSuppStatsEntry 10 }

dot1xSuppLastEapolFrameVersion OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The protocol version number carried in the
        most recently received EAPOL frame."
    REFERENCE
        "<clause ref>, Last EAPOL frame version"
    ::= { dot1xSuppStatsEntry 11 }

dot1xSuppLastEapolFrameSource OBJECT-TYPE
    SYNTAX      MacAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The source MAC address carried in the
        most recently received EAPOL frame."
    REFERENCE
        "<clause ref>, Last EAPOL frame source"
    ::= { dot1xSuppStatsEntry 12 }

-- ---------------------------------------------------------- --
-- IEEE 802.1X MIB - Conformance Information
-- ---------------------------------------------------------- --

dot1xPaeConformance OBJECT IDENTIFIER ::= { ieee8021paeMIB 2 }
```

```
dot1xPaeGroups OBJECT IDENTIFIER ::= { dot1xPaeConformance 1 }

dot1xPaeCompliances OBJECT IDENTIFIER
    ::= { dot1xPaeConformance 2 }

-- ------------------------------------------------------- --
-- units of conformance
-- ------------------------------------------------------- --

dot1xPaeSystemGroup OBJECT-GROUP
    OBJECTS {
        dot1xPaeSystemAuthControl,
        dot1xPaePortProtocolVersion,
        dot1xPaePortCapabilities,
        dot1xPaePortInitialize
    }
    STATUS      deprecated
    DESCRIPTION
        "A collection of objects providing system information
        about, and control over, a PAE."
    ::= { dot1xPaeGroups 1 }

dot1xPaeAuthConfigGroup OBJECT-GROUP
    OBJECTS {
        dot1xAuthPaeState,
        dot1xAuthBackendAuthState,
        dot1xAuthAdminControlledDirections,
        dot1xAuthOperControlledDirections,
        dot1xAuthAuthControlledPortStatus,
        dot1xAuthAuthControlledPortControl,
        dot1xAuthQuietPeriod,
        dot1xAuthSuppTimeout,
        dot1xAuthServerTimeout,
        dot1xAuthMaxReq (deprecated),
        dot1xAuthReAuthPeriod,
        dot1xAuthReAuthEnabled,
        dot1xAuthKeyTxEnabled
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing configuration
        information about an Authenticator PAE."
    ::= { dot1xPaeGroups 2 }

dot1xPaeAuthStatsGroup OBJECT-GROUP
    OBJECTS {
```

```
        dot1xAuthEapolFramesRx,
        dot1xAuthEapolFramesTx,
        dot1xAuthEapolStartFramesRx,
        dot1xAuthEapolLogoffFramesRx,
        dot1xAuthEapolRespFramesRx,
        dot1xAuthEapolReqIdFramesTx,
        dot1xAuthEapolReqFramesTx,
        dot1xAuthInvalidEapolFramesRx,
        dot1xAuthEapLengthErrorFramesRx,
        dot1xAuthLastEapolFrameVersion,
        dot1xAuthLastEapolFrameSource
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing statistics about an
        Authenticator PAE."
    ::= { dot1xPaeGroups 3 }

dot1xPaeAuthDiagGroup OBJECT-GROUP
    OBJECTS {
        dot1xAuthEntersConnecting,
        dot1xAuthEapLogoffsWhileConnecting,
        dot1xAuthEntersAuthenticating,
        dot1xAuthAuthSuccessWhileAuthenticating,
        dot1xAuthAuthTimeoutsWhileAuthenticating,
        dot1xAuthAuthFailWhileAuthenticating,
        dot1xAuthAuthReauthsWhileAuthenticating,
        dot1xAuthAuthEapStartsWhileAuthenticating,
        dot1xAuthAuthEapLogoffWhileAuthenticating,
        dot1xAuthAuthReauthsWhileAuthenticated,
        dot1xAuthAuthEapStartsWhileAuthenticated,
        dot1xAuthAuthEapLogoffWhileAuthenticated,
        dot1xAuthBackendResponses,
        dot1xAuthBackendAccessChallenges,
        dot1xAuthBackendOtherRequestsToSupplicant,
        dot1xAuthBackendAuthSuccesses,
        dot1xAuthBackendAuthFails
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing diagnostic statistics
        about an Authenticator PAE."
    ::= { dot1xPaeGroups 4 }

dot1xPaeAuthSessionStatsGroup OBJECT-GROUP
    OBJECTS {
```

```
        dot1xAuthSessionOctetsRx,
        dot1xAuthSessionOctetsTx,
        dot1xAuthSessionFramesRx,
        dot1xAuthSessionFramesTx,
        dot1xAuthSessionId,
        dot1xAuthSessionAuthenticMethod,
        dot1xAuthSessionTime,
        dot1xAuthSessionTerminateCause
    }
    STATUS      deprecated
    DESCRIPTION
        "A collection of objects providing statistics about the
        current, or last session for an Authenticator PAE."
    ::= { dot1xPaeGroups 5 }

dot1xPaeSuppConfigGroup OBJECT-GROUP
    OBJECTS {
        dot1xSuppPaeState,
        dot1xSuppHeldPeriod,
        dot1xSuppAuthPeriod,
        dot1xSuppStartPeriod,
        dot1xSuppMaxStart,
        dot1xSuppSuppControlledPortStatus,
        dot1xSuppBackendPaeState

    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing configuration
        information about a Supplicant PAE."
    ::= { dot1xPaeGroups 6 }

dot1xPaeSuppStatsGroup OBJECT-GROUP
    OBJECTS {
        dot1xSuppEapolFramesRx,
        dot1xSuppEapolFramesTx,
        dot1xSuppEapolStartFramesTx,
        dot1xSuppEapolLogoffFramesTx,
        dot1xSuppEapolRespFramesTx,
        dot1xSuppEapolReqIdFramesRx,
        dot1xSuppEapolReqFramesRx,
        dot1xSuppInvalidEapolFramesRx,
        dot1xSuppEapLengthErrorFramesRx,
        dot1xSuppLastEapolFrameVersion,
        dot1xSuppLastEapolFrameSource
    }
```

```
    STATUS      current
    DESCRIPTION
        "A collection of objects providing statistics about a
        Supplicant PAE."
    ::= { dot1xPaeGroups 7 }

dot1xPaeAuthSystemGroup OBJECT-GROUP
    OBJECTS {
        dot1xPaeSystemAuthControl,
        dot1xPaePortReauthenticate
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing system information
        about, and control over, an Authenticator PAE."
    ::= { dot1xPaeGroups 8 }

dot1xPaeSystemPortGroup OBJECT-GROUP
    OBJECTS {
        dot1xPaePortProtocolVersion,
        dot1xPaePortCapabilities,
        dot1xPaePortInitialize
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing system information
        about, and control over, a PAE."
    ::= { dot1xPaeGroups 9 }

dot1xPaeAuthSessionStats2Group OBJECT-GROUP
    OBJECTS {
        dot1xAuthSessionOctetsRx,
        dot1xAuthSessionOctetsTx,
        dot1xAuthSessionFramesRx,
        dot1xAuthSessionFramesTx,
        dot1xAuthSessionId,
        dot1xAuthSessionAuthenticMethod,
        dot1xAuthSessionTime,
        dot1xAuthSessionTerminateCause,
        dot1xAuthSessionUserName
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing statistics about the
        current, or last session for an Authenticator PAE."
    ::= { dot1xPaeGroups 10 }
```

```
dot1xPaeAuthInitEapGroup OBJECT-GROUP
    OBJECTS {
        dot1xAuthInitEapCode,
        dot1xAuthInitEapData
    }
    STATUS      current
    DESCRIPTION
        "A collection of objects providing configuration
        information about the initial EAP message generated by
        an Authenticator PAE."
    ::= { dot1xPaeGroups 11 }

-- ------------------------------------------------------------ --
-- compliance statements
-- ------------------------------------------------------------ --

dot1xPaeCompliance MODULE-COMPLIANCE
    STATUS  deprecated
    DESCRIPTION
        "The compliance statement for device support of
        Port Access Control."

    MODULE
        MANDATORY-GROUPS {
            dot1xPaeSystemGroup
        }

        GROUP   dot1xPaeAuthConfigGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Authenticator functions of the PAE."

        OBJECT  dot1xAuthAdminControlledDirections
        SYNTAX  INTEGER {
                    both(0)
                }
        MIN-ACCESS read-only
        DESCRIPTION
            "Support for in(1) is optional."

        OBJECT  dot1xAuthOperControlledDirections
        SYNTAX  INTEGER {
                    both(0)
                }
        DESCRIPTION
```

```
            "Support for in(1) is optional."

        OBJECT dot1xAuthKeyTxEnabled
        MIN-ACCESS read-only
        DESCRIPTION
          "An Authenticator PAE that does not support
          EAPOL-Key frames may implement this object as
          read-only, returning a value of FALSE."

        GROUP   dot1xPaeAuthStatsGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeAuthDiagGroup
        DESCRIPTION
            "This group is optional for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeAuthSessionStatsGroup
        DESCRIPTION
            "This group is optional for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeSuppConfigGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Supplicant functions of the PAE."

        GROUP   dot1xPaeSuppStatsGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Supplicant functions of the PAE."

    ::= { dot1xPaeCompliances 1 }

-- ---------------------------------------------------------- --
-- compliance statements for 802.1aa
-- ---------------------------------------------------------- --

dot1xPaeCompliance2 MODULE-COMPLIANCE
    STATUS  current
    DESCRIPTION
        "The compliance statement for device support of
        Port Access Control."
```

```
    MODULE
        MANDATORY-GROUPS {
            dot1xPaeSystemPortGroup
        }

        GROUP   dot1xPaeAuthSystemGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeAuthConfigGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Authenticator functions of the PAE."

        OBJECT  dot1xAuthAdminControlledDirections
        SYNTAX  INTEGER {
                    both(0)
                }
        MIN-ACCESS read-only
        DESCRIPTION
            "Support for in(1) is optional."

        OBJECT  dot1xAuthOperControlledDirections
        SYNTAX  INTEGER {
                    both(0)
                }
        DESCRIPTION
            "Support for in(1) is optional."

        OBJECT dot1xAuthKeyTxEnabled
        MIN-ACCESS read-only
        DESCRIPTION
            "An Authenticator PAE that does not support
            EAPOL-Key frames may implement this object as
            read-only, returning a value of FALSE."

        GROUP   dot1xPaeAuthStatsGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeAuthDiagGroup
        DESCRIPTION
            "This group is optional for systems that support
            the Authenticator functions of the PAE."
```

```
        GROUP   dot1xPaeAuthSessionStats2Group
        DESCRIPTION
            "This group is optional for systems that support
            the Authenticator functions of the PAE."

        GROUP   dot1xPaeSuppConfigGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Supplicant functions of the PAE."

        GROUP   dot1xPaeSuppStatsGroup
        DESCRIPTION
            "This group is mandatory for systems that support
            the Supplicant functions of the PAE."

        GROUP   dot1xPaeAuthInitEapGroup
        DESCRIPTION
            "This group is optional for systems that support
            the Authenticator functions of the PAE."

    ::= { dot1xPaeCompliances 2 }

END
```

Annex A (normative)

PICS Proforma[1]

<<Throughout this document, all notes such as this one, presented between angle braces, are temporary notes inserted by the Editors for a variety of purposes; these notes will all be removed prior to publication and are not part of the normative text.>>

<<Material borrowed from 802.1D is scattered through this clause as a prompt to the editor and reviewers to supply analagous material for MAC Security, if appropriate.>>

A.1 Introduction

The supplier of a protocol implementation which is claimed to conform to this standard shall complete the following Protocol Implementation Conformance Statement (PICS) proforma.

A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use

a) By the protocol implementor, as a checklist to reduce the risk of failure to conform to the standard through oversight;
b) By the supplier and acquirer—or potential acquirer—of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;
c) By the user—or potential user—of the implementation, as a basis for initially checking the possibility of interworking with another implementation (note that, while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSs);
d) By a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A.2 Abbreviations and special symbols

A.2.1 Status symbols

M	mandatory
O	optional
O.n	optional, but support of at least one of the group of options labelled by the same numeral *n* is required
X	prohibited
pred:	conditional-item symbol, including predicate identification: see A.3.4
¬	logical negation, applied to a conditional item's predicate

A.2.2 General abbreviations

N/A	not applicable
PICS	Protocol Implementation Conformance Statement

[1]*Copyright release for PICS proformas:* Users of this standard may freely reproduce the PICS proforma in this annex so that it can be used for its intended purpose and may further publish the completed PICS.

A.3 Instructions for completing the PICS proforma

A.3.1 General structure of the PICS proforma

The first part of the PICS proforma, implementation identification and protocol summary, is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma is a fixed-format questionnaire, divided into several subclauses, each containing a number of individual items. Answers to the questionnaire items are to be provided in the rightmost column, either by simply marking an answer to indicate a restricted choice (usually Yes or No), or by entering a value or a set or range of values. (Note that there are some items where two or more choices from a set of possible answers can apply; all relevant choices are to be marked.)

Each item is identified by an item reference in the first column. The second column contains the question to be answered; the third column records the status of the item—whether support is mandatory, optional, or conditional; see also A.3.4 below. The fourth column contains the reference or references to the material that specifies the item in the main body of this standard, and the fifth column provides the space for the answers.

A supplier may also provide (or be required to provide) further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labelled *Ai* or *Xi,* respectively, for cross-referencing purposes, where *i* is any unambiguous identification for the item (e.g., simply a numeral). There are no other restrictions on its format and presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformation Statement for the implementation in question.

NOTE—Where an implementation is capable of being configured in more than one way, a single PICS may be able to describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering some subset of the implementation's configuration capabilities, in case that makes for easier and clearer presentation of the information.

A.3.2 Additional information

Items of Additional Information allow a supplier to provide further information intended to assist the interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS can be considered complete without any such information. Examples might be an outline of the ways in which a (single) implementation can be set up to operate in a variety of environments and configurations, or information about aspects of the implementation that are outside the scope of this standard but that have a bearing upon the answers to some items.

References to items of Additional Information may be entered next to any answer in the questionnaire, and may be included in items of Exception Information.

A.3.3 Exception information

It may occasionally happen that a supplier will wish to answer an item with mandatory status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No pre-printed answer will be found in the Support column for this: instead, the supplier shall write the missing answer into the Support column, together with an *Xi* reference to an item of Exception Information, and shall provide the appropriate rationale in the Exception item itself.

An implementation for which an Exception item is required in this way does not conform to this standard.

NOTE—A possible reason for the situation described above is that a defect in this standard has been reported, a correction for which is expected to change the requirement not met by the implementation.

A.3.4 Conditional status

A.3.4.1 Conditional items

The PICS proforma contains a number of conditional items. These are items for which both the applicability of the item itself, and its status if it does apply—mandatory or optional—are dependent upon whether or not certain other items are supported.

Where a group of items is subject to the same condition for applicability, a separate preliminary question about the condition appears at the head of the group, with an instruction to skip to a later point in the questionnaire if the "Not Applicable" answer is selected. Otherwise, individual conditional items are indicated by a conditional symbol in the Status column.

A conditional symbol is of the form "**pred:** S" where **pred** is a predicate as described in A.3.4.2 below, and S is a status symbol, M or 0.

If the value of the predicate is true (see A.3.4.2), the conditional item is applicable, and its status is indicated by the status symbol following the predicate: the answer column is to be marked in the usual way. If the value of the predicate is false, the "Not Applicable" (N/A) answer is to be marked.

A.3.4.2 Predicates

A predicate is one of the following:

a) An item-reference for an item in the PICS proforma: the value of the predicate is true if the item is marked as supported, and is false otherwise;
b) A predicate-name, for a predicate defined as a boolean expression constructed by combining item-references using the boolean operator OR: the value of the predicate is true if one or more of the items is marked as supported;
c) A predicate-name, for a predicate defined as a boolean expression constructed by combining item-references using the boolean operator AND: the value of the predicate is true if all of the items are marked as supported;
d) The logical negation symbol "¬" prefixed to an item-reference or predicate-name: the value of the predicate is true if the value of the predicate formed by omitting the "¬" symbol is false, and vice versa.

Each item whose reference is used in a predicate or predicate definition, or in a preliminary question for grouped conditional items, is indicated by an asterisk in the Item column.

A.4 PICS proforma for IEEE Std 802.1AE

A.4.1 Implementation identification

Supplier	
Contact point for queries about the PICS	
Implementation Name(s) and Version(s)	
Other information necessary for full identification—e.g., name(s) and version(s) of machines and/or operating system names	
NOTE 1—Only the first three items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification. NOTE 2—The terms Name and Version should be interpreted appropriately to correspond with a supplier's terminology (e.g., Type, Series, Model).	

A.4.2 Protocol summary, IEEE Std 802.1AE

Identification of protocol specification	IEEE Std 802.1AE, Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Security
Identification of amendments and corrigenda to the PICS proforma which have been completed as part of the PICS	Amd. : Corr. : Amd. : Corr. :
Have any Exception items been required? (See A.3.3: the answer Yes means that the implementation does not conform to IEEE Std 802.1AE.)	No [ ] Yes [ ]

Date of Statement	

A.5 Major Capabilities

Item	Feature	Status	References	Support
MACP	Does the implementation provide the MAC Service, as specified in <ref>, for use by end system functionality in the containing system?	M	A.6	
EISSP	Does the implementation provide the Extended Internal Sublayer Service as specified in IEEE Std 802.1Q to support the MAC Bridge functionality?	O	A.7	
EISSU	Is each specific MAC Technology used as specified by IEEE Std 802.1Q for the support of the MAC Extended Internal Sublayer Service for that MAC Technology? (The PICS Proforma(s) required by IEEE Std 802.1Q shall also be completed.) (If support of a specific MAC technology is claimed any PICS Proforma(s) required by the Standard specifying that technology shall also be completed.)	M	IEEE Std 802.1Q 6.4, . A.8	Yes [ ]
SECS	Does the implementation support the full range of security services specified in Clause 6 of this standard?	M	<<ref>> A.9	
EX1	Does the implemention provide this mandatory major capability?	M	<<ref>> A.10	Yes [ ]
EX2	Is this major capability supported?	O	<<ref>> A.11	Yes [ ] No [ ]
EX3	Does the implementation do what is supposed to do in respect of this major capability?	**EX2:**M	<<ref>> A.12	Yes [ ] N/A[ ]

A.6 Provision of the MAC Service

Item	Feature	Status	References	Support
MACP-1	Has it been done right?	M		Yes [ ]
MACP-2	First detail?	M		Yes[ ]
MACP-3	Second detail?	M	6.4, .	Yes [ ]
MACP-4	Are the MAC status parameters implemented on all Ports?	M	6.4, .	Yes [ ]

Predicates:
GOOK= GOOK_SPEC[Yes]

A.7 Provision of the Extended Internal Sublayer Service

Item	Feature	Status	References	Support
EISSP-1	Has it been done right?	**EISSP:**M		Yes [ ]
EISSP-2		M		Yes[ ]

A.8 Use of the Extended Internal Sublayer Service

Item	Feature	Status	References	Support
EISSU-1				
EISSU-2				

A.9 Security services

Item	Feature	Status	References	Support
SECS-1				Yes [ ]
SECS-2				Yes [ ]

A.10 Major Capability 1

Item	Feature	Status	References	Support
EX1-1				Yes [ ]
EX1-2				Yes [ ]

A.11 Major Capability 2

Item	Feature	Status	References	Support
EX2-1				Yes [ ]
EX2-2				Yes [ ]

A.12 Major Capability 3

Item	Feature	Status	References	Support
EX3-1				Yes [ ]
EX3-2				Yes [ ]

Annex Y (informative)

Bibiography

<<Items in the References clause should only be those that are definitely referenced by the document, not just useful background reading. The latter should go here.>>

Annex Y (informative)

Draft Changes

ch 1.2 added scope verbatim from PAR

ch 1.4 Definitions - note to fill in

ch. 6 typo, "the EISS is derived from the ISS"

added annex X to keep changes, will throw away

Annex Z (informative)

Commentary

<<Editor's Note: This is a temporary Annex, included as a record of technical issues and their disposition. This annex will be removed prior to Sponsor Ballot, and preserved on the 802.1 web site for future reference[1].>>

<<The order of discussion of issues is intended to help the reader understand first what is the draft, secondly what may be added, and thirdly what has been considered but will not be included. In pursuit of this goal, issues where the proposed disposition is "no change" will be moved to the end. The description of issues is updated to reflect our current understanding[2] of the problem and its solution: where it has been considered useful to retain the original comment, in whole or part, either to ensure that its author does not feel that it has not been sufficiently argued or the editor suspects there may be further aspects to the issue, that has been done as a footnote.>>

Z.1 Replay protection

Notwithstanding the emphasis on “connectionless” confidentiality and integrity in the PAR, and the implication that not only is the service provided connectionless (i.e. one request primitive has not relationship to any other except for quality of service aspects) but also the service support is connectionless, i.e. there is no relationship between one frame and another, it has been agreed that the discussion, and potential provision, of replay protection falls within the scope of the PAR.

Replay protection is provided through including a sequence number, such as the Packet Number PN, in the frame header. Determining whether this PN has been received before is usually done in one of two ways. First by keeping a list of numbers that have been seen. Alternatively, an upper and lower window size could be kept, and any packet outside the window is dropped. If the packet is in the window, it is checked to see whether it was previously received or not.

The PNs cannot be changed without detection because they are protected by the ICV.

Z.1.1 Disposition

And this is what we have decided so far....

Z.2 Cryptographic suites

Crypto choices include:

1) parameters: - what are they and what are their sizes- PN (packet number) length, ICV length,.
2) privacy, confidentiality modes: CTR, CBC,.
3) integrity modes: HMAC-SHA1, MD5, OMAC, PMAC,...
4) encryption algorithms, block ciphers: AES-128, DES, 3DES-EDE,.
5) combination modes: CCM, OCB, CWC, EAX

[1]The footnotes in this annex provide further background to its development. Most of the highly subjective material, who said what and were they were right etc. together with temporary notes on blind alleys will be put into the footnotes so that they can be easily stripped out when the final annex is preserved.

[2]This annex is not intended therefore to be a complete historical record of the development of the draft. The formal record is largely captured in the Disposition of Comments on each ballot.

Z.2.1 Cipher suites

It is better to use cipher suites rather than to choose individual algorithms to match together to provide different functions, such as, confidentiality, integrity, key management, and replay protection. The algorithms within a given cipher suite are well-understood to work together, and the set of algorithms is verifiability secure for some set of configured parameters. On the contrary, to not use a cipher suite would open the choices for providing key management, integrity and confidentiality to an arbitrary combination of algorithms and parameter settings, which are not necessarily known to work well together and be secure. For example, past experience suggests that using modes differently than in their default configurations has led to poor security.

Here is a suggested Table of cipher suites for confidentiality.

Table 1—Suggested Confidentiality Ciphers

EUI-48	Cipher #	Type	Mandatory/ Optional	Defined in
XX-XX-XX	0	NULL	Mandatory	x.y.z
XX-XX-XX	1	AES-128 in CCM Mode	Mandatory	
XX-XX-XX	2	AES-128 in OCB Mode	Optional	
XX-XX-XX	3	OMAC	M	
XX-XX-XX	4	PMAC	O	
XX-XX-XX	5	CWC	M	
XX-XX-XX	...	TBD		
XX-XX-XX	...-32767	Reserved		
XX-XX-XX	32768-65535	Playpen		
XX-XX-XX	0-65535	Vendor Propri-etary		

A Null encryption cipher suite is necessary, see Cl 8.

It is argued that a minimal number of cipher suites should be mandatory, while the rest are optional. Optional cipher suites might be mandatory for some devices for technical reasons, e.g., parallelizability. It is necessary to make sure the dividing line is well-defined. Define the dividing line, e.g., OCB is mandatory above 1.1. Gb/ps.

Sets of ciphers for port authentication and for key exchange are also necessary, but will be under another PAR.

Z.2.2 Crypto issues

Parameter values: Parameter values need to be constant during the life of an Security Association SA. Parameter values that can change dynamically create vulnerabilities. Rather than attacking the cryptographic algorithm, an attacker can attack the negotiation, so that a weaker type of cryptography is chosen, which the attacker can then break more easily. For example, in wireless, an attacker could cause WEP to be chosen rather than something more secure. Therefore, once parameters values have been established, there should not be a mechanism for changing them.

Should parameter values be negotiable? There can be reasons for making some parameters negotiable at setup time. For example, PN length might be negotiated because different priorities exist for different MAC/ PHYs. One way to incorporate this need is by including the various options within the set of defined cipher suites. The way to provide for different parameter values is through different cipher suites -- for example one might have 128-bit value and another 256-bit value. Using cipher suites in this way is a simple way to allow policy.

The crypto needs of different media types are different -speed, cost, processing power needs. Provider bridges require that different media types interact, on both ends of an SA. How should the differing crypto needs be treated?

Security establishment: There is a chicken and egg situation that needs to be considered on a MAC specific basis. How does a link get established if it needs to be secure to be established and you can only do security on an established link? Which parts of the frame are protected and which are not? Must avoid setting up a race condition by design. For 802.3, the decision is to apply security transform only to data, not to control frames

Z.2.3 Observations on cryptographic algorithms

1) AES is the block cipher du jour
2) RC4-40 used for exportability, but is not good for engineering reasons, as it has a heavily serial algorithm
3) DES is deprecated (by FIPS) for new equipment
4) CTR (privacy mode) is parallelizable
5) FIPS has not yet chosen an integrity mode. It is thought likely to choose OMAC, which is not parallelizable
6) integrity modes that are parallelizable are often encumbered - PMAC,...
7) it is possible to use an authentication specific algorithm, HMAC-SHA1. However it requires independent hardware
8) combo mode CCM- not parallelizable, not encumbered, used in 802.11i
9) combo mode OCB - parallelizable, encumbered, requires more gates to do AES decrypting
10) combo mode CWC - parallelizable, not encumbered,...

Z.3 Vulnerabilities

IEEE 802 technologies are vulnerable at the interfaces between L2 and L3. Vulnerability is created where there is loose coupling between layers, where there is difference of address spaces, where there is security at layer n-1 but not layer n. ARP spoofing occurs because of the need for an unprotected discovery protocol to discover the lower layer address. MACsec probably cannot offer any help.

Z.3.1 ARP Spoofing

We cannot offer protection against disclosure due to ARP spoofing, in which an attacker sends gratuitous ARP messages claiming to have IP addresses of other stations. The attacker can then intercept, read, and alter messages between any two points in a point to point topology. The ICV will be recalculated and the altered data will pass integrity check. If the message is cryptographically protected above L2, for example with IPsec, the data cannot be read or changed by the attacker. This threat occurs at the intersection between the L2 and L3 layers.

There is no protection against legitimate users - in ARP spoofing, the attacker has legitimate use of the LAN. It is a case of legitimate user with bad intent. MACsec does offer the ability to identify the bad user. If the bad behavior is unintentional, for example, due to misconfiguration, it can be corrected, but if it is unintentional, then MACsec only identifies the attacker, but cannot prevent the attack.

The ICV is recomputed every time the frame presented to the MAC layer, so a man-in-the-middle can make unauthorized changes and it will pass the integrity check. Thus, we have point to point integrity, but not global integrity.

Z.4 Parameters and Frame Format

Z.4.1 Requirements

What should the parameters be bound to? A cipher suite.

If the parameters are variable, then the frame formatter might need to behave dynamically, redefine MTU, etc.

Parameter choices

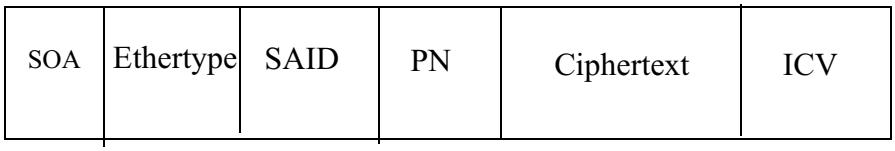
1) A sequence counter - Packet Number (PN), Initial Value (IV) - how long?
2) An integrity check - Message Integrity Code, MIC, Message Authentication Code MAC, Integrity Check Value (ICV) - length?
3) What kind of ciphertext?
4) How much of packet should be in ciphertext?
5) Security Association ID (SAID)

Different media technologies give rise to different cryptographic needs, such as length of the PN and whether parallelizability is necessary. This arises in the context of Provider Bridges, which may result in end-to-end connections between dissimilar technologies. One suggestion is that we should have a global default set of parameters to be present on all devices. This however creates challenges to meet the criteria of both technologies, such as needing speed (802.3) and also requiring low cost (802.11).

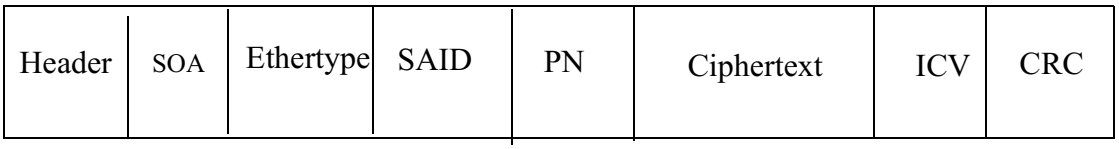
Vendor proprietary area and playpen areas, with appropriate EUI-48s should be included.

Z.4.2 Frame

The frame is in the stack below the security layer. Here is a possible frame layout.



If the SecY is the last thing above the media specific piece, then the frame format would be:



Z.4.3 Header data

The MAC level header fields are not in MSDU, but are in the MPDU, except in the one case shown above in Figure.., where the SecY is the... The header fields are defined by the various IEEE 802 MAC/PHY specifications. Thus they are different for different IEEE 802 technologies.

Header data consists of authenticated parts, covered by the ICV, and non-authenticated parts, not covered by the ICV. Not all fields in the header require authentication. See Figure 8.1

Placement: Ideally the header data would be placed after the PN just before the ciphertext in order to allow some time between nonce construction and data encryption.

Not all fields in the header require authentication. For example, the Header Check Sequence (HCS) in IEEE 802.16 would not be authenticated as the field is itself computed from the first 5 bytes of the header in order to discover bit errors. In fact, it would prove problematic in implementation to authenticate. The HCS is generated after authentication has been done. Another example is from 802.11. The upper four bits of the sequence control field, SC, cannot be authenticated for implementation reasons; the value of the bits is known only a moment before transmission.

Z.4.4 Secure Origin Authenticity (SOA)

Source and destination addresses, SA and DA, in the MPDU are the original addresses, the end to end addresses. In a multi-point or Provider Bridge network,. the SA and DA are not the addresses of intermediate devices that are doing the encryption and decryption. The SecY is part of the bridge stack, and thus it’s address is not seen. It would be useful in the PB case in particular, to make available the knowledge of where the encryption was applied. This is what the SOA provides. It is the address of the Bridge port, which is used both by the Bridge and the SecY, called a Common Port, see Cl 8.

Z.4.5 Ethertype

16 bit ethertype field.

Z.4.6 SecTAG

Concatenation of the SOA, the Ethertype and the SAID.(?)

Z.4.7 Security Association Identification (SAID)

Placement: The SAID should occur as early as possible in the frame to allow early retrieval of keys.

How many SAs should be supported? Sixteen bits supports 65536 SAs. Separate SAs are needed for broadcast/multicast groups. Will 8 bits work? 802.16 supports 256 SAs.

There are two possible sources of unique numbering - between the endpoints; a number space for endpoints only. Or, a unique number space within a domain of SAIDs. In order for the function of determining who encapsulated the packet, the ID number must be unique among a group of devices, not just unique between a pair of devices.

We need to allow optimizations in which the SAID does not show up in frames. For, example, the SAID is not needed in a point to point link. It may be possible to negotiate whether the SAID will be not included in frames, where bandwidth is scarce and SAID unnecessary. It should be in the frame architecturally, though in implementation it may be compressed out, if its value is always the same.

Z.4.8 Packet Number (PN)

Placement: The packet number should go as early as possible in the frame because a nonce needs to be constructed before any cryptographic operations can happen.

Size: The PN strength is measured in the time to re-key, which is directly dependent on the length of the PN. Therefore, longer is better than shorter. The time to re-key is inversely proportional to the packet rate.

There is a discrepancy in what is a good PN size for different cases, there is no good PN size for all cases. Compare the slowest IEEE 802 technology with the fastest to see the magnitude of the difference between optimal in each case. The minimum rate is approximately 10kb/ps (IEEE 802.15), maximum rate approximately 100 Gb/ps (arbitrary optical medium). There is a 10^7 difference, which is approximately 23.5 bits. so the slowest links will want 24 fewer bits to achieve the same re-keying rate as the fastest link will want for optimal operation. The gap will only widen in the future.

How can this issue of choosing the PN size be resolved? A number of options were considered:

1) Pick suitable case for highest packet rate interface. Make it 8 bytes. The problem here is that then slower devices suffer, which may have as a consequence poor adoption rate.
2) Tie the PN size to the PHY type. Provider bridges span between different PHY types, so this option won't work.
3) Variable length encoding. The PN size could grow with the PN magnitude. This is expensive to do in hardware implementation. The savings is not great because most occupancy is caused by large sized values.
4) Variable PN size, coded in the header. This option has bad security implications. An attacker spoofing the packet can make the PN size smaller and thereby easier to crack.

5) Variable PN size negotiated between stations during setup. Negotiated at the time of SA formation and remain constant for the lifetime of the SA.

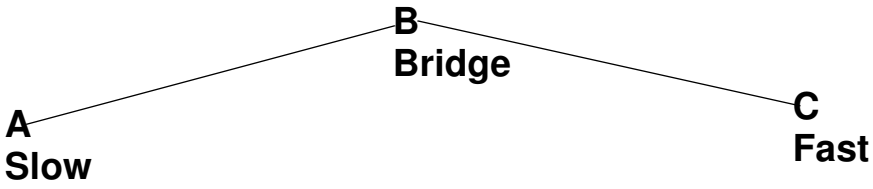
Agreement to negotiate a priori for the life of the SA. The PN size should be a parameter in the cipher suite, so that two sizes create two cipher suites. It was decided to chose the PN sizes to differ maximally, the largest and the smallest - either 64 bits or 24 bits. The rationale is that it is usually desirable to either maximize security with a large size, or to minimize the damaging effect on throughput with the smaller size.

For a given technology, the PN size should have a default value based on the maximum packet rate. Therefore, in the case where there is not a Provider Bridge, the negotiation will be brief, both sides agreeing on the default.

Provider Bridges causes some complexity with respect to the PN size. What about negotiating PN size in the case of Provider Bridges? If the PN size is by default the smaller PN choice of the two ends, then the slower device limits the maximum packet rate.

Negotiating the size of the PN in the Provider Bridge case causes a vulnerability. Attackers can attack the negotiation rather than the cryptography, in order to get the PN size smaller and thus make the cryptography easier to crack. Requirements for cryptography should include assurance that the negotiation leads to a chosen size no smaller than the minimum acceptable to the slower device.

With Provider Bridges a vulnerability can be caused. A and C negotiate a small PN in accordance with A, the slower device. An active attacker on the fast side of B is capable of breaking a small PN. If the slower transmission rate is used in security assumptions, they become invalid if the link from B to C has a faster data rate.



Negotiating the PN size as a parameter for a particular algorithm is not a good idea. The negotiation might lead to the improper use of a cryptographic function. For example, a security proof may assume that the PN size is a particular value.

Choosing between cipher suites would work since we are limited to known good cases. Assume that each cipher suite configuration had been validated.

Negotiating PN size exposes the existence of a Provider Bridge PB. If there is not a PB, then there is a single medium and the default PN will quickly be established, so any negotiation between stations exposes the fact that they are separated by a PB.

Two proposals for PN negotiation:

1) When operating across Provider Bridges, the slower device yields to the faster device PN length and suffers throughput drop. This minimizes security risk on the fast side, and makes impersonating a Provider Bridge pointless, since the more secure length is chosen.

When on a non-provider Bridge link, negotiation leads to the default for that MAC/PHY.

2) When operating across Provider Bridges, the faster device yields to the slower device PN length and suffers the potential increase in re-keying rate. This prevents imposing an undue overhead on a slow link.

When on a non-provider Bridge link, negotiations leads to the default for that MAC/PHY, or to a shared enhanced mode value.

<<example of two cipher suites that differ only in a parameter value>>

Z.4.9 Ciphertext

Placement: The placement of the ciphertext in the frame is similar to the placement of the plaintext.

Z.4.10 Integrity Check Value (ICV)

When to compute the ICV: Is the ICV going to be computed over encrypted data, or cleartext? A big issue to consider. Although this will be decided as part of a cipher suite choice, we should understand the implications. Computing the ICV over encrypted data makes it possible for the ICV to be okay, but the data to be meaningless.

Intuitively, you would do the ICV over cleartext, but there are significant advantages to computing it over encrypted text. If it is computed over encrypted data, then it allows rapid recognition that the packet does not pass the ICV. Such rapid recognition is paramount in thwarting DOS attacks. On the other hand, if the ICV is on cleartext, and then the frame is encrypted, then you have the cost of decryption prior to computing the ICV, which takes considerable time. If the ICV is over encrypted text, then you first compute the ICV, so you learn quickly if the frame is not authentic.

The principal issue involved in whether to encrypt before or after computing the ICV is the trade-off between the cost of computing the ICV value and the cost of decrypting. Since these vary for different algorithms, different cipher suites will differ in when they compute the ICV.

Placement: It is easier for implementation to place the ICV at the end of the frame for both transmission and reception. On transmission, if the ICV is placed at the end of the packet, it can be computed on the fly. On reception, if the ICV is at the end of the packet, it does not need to be stored during ICV computation.

Size: Some modes are susceptible to birthday attacks and some modes are not susceptible to such offline attacks. For example HMAC SHA1, as a digest, is subject to a birthday attack, because the ICV is in the clear, whereas CCM is not vulnerable to a birthday attack because the ICV is encrypted.

For a mode that is susceptible to an offline attack, the crypto strength is proportional to sqrt(2^n), where n is the number of bits in the ICV. A mode that is not susceptible to birthday attack has crypto strength proportional to 2^n. For a given level of security, the ICV for a susceptible mode needs to be twice as long as the ICV for a non-susceptible mode. In a birthday attack, the number of cases that need to be examined before finding a match is the square root of the number of states of the ICV.

Thus if we avoid modes susceptible to birthday attacks, the size of the ICV can be at least less than half the length for the same strength.

Another consideration with respect to the size of the ICV, is that it is difficult computationally to do a large number of guesses, 2^80, say. If a mode is not subject to offline attacks, it forces the attacker to launch an active attack, which is more likely to be detected.

The implication of this discussion is that we should choose modes that are not subject to birthday attacks because then the ICV can be half as long. This is particularly an advantage for media technologies for whom the longer ICV uses up scarce bandwidth resources, e.g., wireless 802.11 and 802.16.

Z.4.11 Frame Expansion

Due to additional fields, frame expansion is required and needs to be considered by 802.3. Theoretically, there are 3 alternatives: expansion, fragmentation, MTU limitation. Practically, only expansion is acceptable. We are asking 802.3 for a larger frame size.

An alternative to fragmentation is to limit the MTU size, thereby forcing the higher layer protocol to fragment the packet. In the wireless domain this requirement is not an issue since such MTU limitation is normal. An MTU of any size is accepted at the MAC layer and is fragmented later. However some wired standards do not have a means of signaling MTU variation to the next higher layer.

The previous IEEE 802.10 specification fragmented the packet into two when MTO adjustment was not supported. Fragmentation by the security protocol is not natural function for a security sublayer, should be done by media access control as media need and dictate.

It is felt that there is a need for a generic service to propagate MTU size upwards.