INTERNATIONAL STANDARD

ISO/IEC/IEEE 8802-1X

First edition
2013-12-01

AMENDMENT 1
2016-02-15

Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks —

Part 1X: Port-based network access control

AMENDMENT 1: MAC security key agreement protocol (MKA) extensions

Technologies de l'information — Télécommunications et échange d'information entre systèmes — Réseaux locaux et métropolitains — Exigences spécifiques —

Partie 1X: Contrôle d'accès au réseau basé sur le port

AMENDEMENT 1: Extensions du protocole d'accord de clés de sécurité MAC

Reference number
ISO/IEC/IEEE 8802-1X:2013/AMD 1:2016(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat, the IEC Central Office and IEEE do not accept any liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies and IEEE members. In the unlikely event that a problem relating to it is found, please inform the ISO Central Secretariat or IEEE at the address given below.

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
Case postale 56
CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland
E-mail inmail@iec.ch
Web www.iec.ch

Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York
NY 10016-5997, USA
E-mail stds.ipr@ieee.org
Web www.ieee.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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.

The main task of ISO/IEC JTC 1 is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is called to the possibility that implementation of this standard may require the 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. ISO/IEEE is not responsible for identifying essential patents or patent claims for which a license may be required, for conducting inquiries into the legal validity or scope of patents or patent claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance or a Patent Statement and Licensing Declaration Form, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from ISO or the IEEE Standards Association.

Amendment 1 to ISO/IEC/IEEE 8802-1X:2013 was prepared by the LAN/MAN of the IEEE Computer Society (as IEEE 802.1Xbx-2014). It was adopted by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*, in parallel with its approval by the ISO/IEC national bodies, under the “fast-track procedure” defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE. IEEE is responsible for the maintenance of this document with participation and input from ISO/IEC national bodies.

(blank page)

IEEE STANDARDS ASSOCIATION

IEEE Standard for
Local and metropolitan area networks—

Port-Based Network Access Control

Amendment 1: MAC Security Key Agreement Protocol (MKA) Extensions

IEEE Computer Society

Sponsored by the
LAN/MAN Standards Committee

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

IEEE Std 802.1Xbx™-2014
(Amendment to
IEEE Std 802.1X™-2010)

IEEE Std 802.1Xbx™-2014
(Amendment to
IEEE Std 802.1X™-2010)

IEEE Standard for
Local and metropolitan area networks—

Port-Based Network Access Control

Amendment 1: MAC Security Key Agreement Protocol (MKA) Extensions

Sponsor

LAN/MAN Standards Committee
of the
IEEE Computer Society

Approved 10 December 2014

IEEE-SA Standards Board

Abstract: Media Access Control security (MACsec) Key Agreement protocol (MKA) data elements and procedures that provide additional security and manageability capabilities, including the ability to maintain secure communication while the operation of MKA is suspended, when used in conjunction with MACsec Cipher Suites that support Extended Packet Numbering are added in this amendment.

Keywords: authorized port, confidentiality, data origin authenticity, IEEE 802.1X™, IEEE 802.1Xbx™, integrity, LANs, local area networks, MAC Bridges, MAC security, MAC Service, MANs, metropolitan area networks, port based network access control, secure association, security, transparent bridging

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: **ISBN 978-0-7381-9435-6** **STD20045**
Print: **ISBN 978-0-7381-9436-3** **STDPD20045**

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

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

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. 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.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive 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 comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

> Secretary, IEEE-SA Standards Board
> 445 Hoes Lane
> Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, 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 order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp or contact IEEE at the address listed previously. For more information about the IEEE SA or IEEE's standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

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 by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at http://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this amendment was submitted to the IEEE-SA Standards Board for approval, the IEEE 802.1 Working Group had the following membership:

Glenn Parsons, *Chair*
John Messenger, *Vice Chair*
Mick Seaman, *Security Task Group Chair, Editor*

Ting Ao
Christian Boiger
Paul Bottorff
David Chen
Feng Chen
Weiying Cheng
Diego Crupnicoff
Rodney Cummings
Patrick Diamond
Aboubacar Kader Diarra
Janos Farkas
Norman Finn
Geoffrey Garner
Anoop Ghanwani
Mark Gravel
Eric W. Gray
Craig Gunther
Stephen Haddock

Hitoshi Hayakawa
Jeremy Hitt
Rahil Hussain
Tony Jeffree
Michael Johas Teener
Peter Jones
Hal Keen
Marcel Kiessling
Yongbum Kim
Philippe Klein
Jouni Korhonen
Jeff Lynch
Ben Mack-Crane
Christophe Mangin
James McIntosh
Eric Multanen
Donald Pannell

Karen Randall
Maximilian Riegel
Dan Romascanu
Jessy V. Rouyer
Panagiotis Saltsidis
Behcet Sarikaya
Daniel Sexton
Johannes Specht
Kevin B. Stanton
Wilfried Steiner
Vahid Tabatabaee
Patricia Thaler
Jeremy Touve
Karl Weber
Yuehua Wei
Brian Weis
Jordon Woods
Juan-Carlos Zuniga

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Thomas Alexander
Butch Anton
Olugbenga Ayinde
William Byrd
Juan Carreon
Keith Chow
Charles Cook
Grazia Delia
Sourav Dutta
Richard Edgar
Yukihiro Fujimoto
Devon Gayle
Gregory Gillooly
Randall Groves
Michael Gundlach
Werner Hoelzl
Atsushi Ito

Tony Jeffree
Peter Jones
Shinkyo Kaku
Piotr Karocki
Stuart Kerry
Max Kicherer
Jeff Koftinoff
Bruce Kraemer
Yasushi Kudoh
Thomas Kurihara
Paul Lambert
Hyeong Ho Lee
Shen Loh
Elvis Maculuba
Jouni Malinen
Michael Newman
Nick S.A. Nikjoo

Satoshi Obara
Satoshi Oyama
Karen Randall
Maximilian Riegel
Jessy V. Rouyer
Mick Seaman
Kapil Sood
Thomas Starai
Rene Struik
Walter Struppler
Joseph Tardo
William Taylor
Patricia Thaler
Dmitri Varsanofiev
Hung-Yu Wei
Brian Weis
Oren Yuen
Daidi Zhong

When the IEEE-SA Standards Board approved this amendment on 10 December 2014, it had the following membership:

John Kulick, *Chair*
Jon Walter Rosdahl, *Vice Chair*
Richard H. Hulett, *Past Chair*
Konstantinos Karachalios, *Secretary*

Peter Balma
Farooq Bari
Ted Burse
Clint Chaplain
Stephen Dukes
Jean-Phillippe Faure
Gary Hoffman
Michael Janezic
Jeffrey Katz
Joseph L. Koepfinger*
David J. Law
Hung Ling
Oleg Logvinov
Ted Olsen
Glenn Parsons
Ron Peterson
Adrian Stephens
Peter Sutherland
Yatin Trivedi
Phil Winston
Don Wright
Yu Yuan

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*

Catherine Berger
IEEE-SA Content Production and Management

Kathryn Bennett
Program Manager, IEEE-SA Technical Program Operations

Introduction

This introduction is not part of IEEE Std 802.1Xbx™-2014, IEEE Standard for Local and metropolitan area networks—Port-Based Network Access Control—Amendment 1: MAC Security Key Agreement Protocol (MKA) Extenstions.

This first amendment to IEEE Std 802.1X-2010, extends MKA to realize additional security and manageability capabilities made possible by the IEEE Std 802.1AEbw™ amendment that added extended packet numbering Cipher Suites to IEEE Std 802.1AE™-2006. Secure connectivity association (CA) members can now temporarily suspend MKA operation without causing protocol timeouts that would disrupt secure data transfer, thus allowing in-service control plane software upgrades.

The first edition of IEEE Std 802.1X was published in 2001. The second edition, IEEE Std 802.1X-2004 clarified areas related to mutual authentication and the interface between IEEE 802.1X specified state machine, and those specified by the Extensible Authentication Protocol (EAP), and by IEEE Std 802.11™ in support of IEEE Std 802.1X.

The third edition, IEEE Std 802.1X-2010, added authenticated key agreement in support of IEEE Std 802.1AE™ MAC Security, clarifying and generalizing the relationship between the common architecture specified for port-based network access control, and the functional elements and protocols that support that architecture as specified in IEEE Std 802.1X, other IEEE 802® standards, and in IETF RFCs. Further changes updated the standard to reflect best current practice, insisting, for example, upon mutual authentication methods and using such methods in examples. A greater emphasis was placed on the security of systems accessing the network, as well as upon the security of the network accessed, and some prior provisions, with a more comprehensive treatment of segregating and limiting connectivity to unauthenticated systems. Applications of port-based network access that use IEEE Std 802.1AE MAC Security (MACsec) and/or MKA (MACsec Key Agreement protocol) are described.

Every effort was made to ensure that systems conformant to IEEE Std 802.1X-2010 will interoperate, without prior configuration, with implementations conforming to IEEE Std 802.1X-2004 and IEEE Std 802.1X-2001. However it is anticipated that claims of conformance in respect of some existing implementations, not needing to support IEEE Std 802.1AE and already conforming to best current practice as of 2010, will continue to refer to IEEE Std 802.1X-2004. IEEE Std 802.1X-2010 includes a number of improvements to the specification of the port access control protocol (PACP) state machines and their relationship to EAP methods and state machines.

Contents

Figures

Tables

IEEE Standard for Local and metropolitan area networks—

Port Based Network Access Control

Amendment 1: MAC Security Key Agreement Protocol (MKA) Extensions

[This amendment is based on IEEE Std 802.1X™-2010.]

NOTE—The editing instructions contained in this amendment define how to merge the material contained therein into the existing base standard and its amendments to form the comprehensive standard.

The editing instructions are shown in ***bold italic***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strikethrough~~ (to remove old material) and underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Deletions and insertions may require renumbering. If so, renumbering instructions are given in the editing instruction. ***Replace*** is used to make changes in figures or equations by removing the existing figure or equation and replacing it with a new one. Editing instructions, change markings, and this NOTE will not be carried over into future editions because the changes will be incorporated into the base standard.

IIMPORTANT NOTICE: IEEE Standards documents are not intended to ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/IPR/disclaimers.html.

2. Normative references

Change the Normative references clause as follows:

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 802.1D™, IEEE Standard for Local and Metropolitan Area Networks: Media access control (MAC) Bridges.[1, 2]

IEEE Std 802.1Q™, IEEE Standard for Local and Metropolitan Area Networks: Bridges and Bridged Networks ~~Virtual Bridged Local Area Networks~~.

IEEE Std 802.1AB™, IEEE Standard for Local and Metropolitan Area Networks: Station and Media Access Control Connectivity and Discovery.

~~IEEE Std 802.1ad™-2005, IEEE Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks—Amendment 4: Provider Bridges.~~

IEEE Std 802.1AE™, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security.

IEEE Std 802.1AE™-2006, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security.[3]

IEEE Std 802.1AEbn™-2011, IEEE Standard for Local and Metropolitan Area Network — Media Access Control (MAC) Security — Amendment 1: Galois Counter Mode—Advanced Encryption Standard—256 (GCM–AES–256) Cipher Suite.

IEEE Std 802.1AEbw™-2013, IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (MAC) Security — Amendment 2: Extended Packet Numbering.

IEEE Std 802.1AX™, IEEE Standard for Local and Metropolitan Area Networks: Link Aggregation.

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.[4]

IEEE Std 802.3™, IEEE Standard for Ethernet~~Information technology—Local and metropolitan area networks—Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications~~.

[1] IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA. IEEE publications can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org.

[2] The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.

[3] This standard refers to the latest edition of IEEE Std 802.1AE in addition to referencing specific revisions and amendments.

[4] 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, Piscataway, NJ 08854, USA. ISO [IEEE] and ISO/IEC [IEEE] documents can be ordered on-line from the IEEE Standards Website: http://www.standards.ieee.org.

IEEE Std 802.11™, IEEE Standard for 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.

IEEE Std 802.17™-2004 IEEE Standard for Information Technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 17: Resilient packet ring (RPR) access method and physical layer specifications.

IEEE Std 802.1AR™, IEEE Standard for Local and Metropolitan Area Networks: Secure Device Identifier.

IETF RFC 2578, STD 58, Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2), McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.[5]

IETF RFC 2579, STD 58, Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2), McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.

IETF RFC 2580, STD 58, Conformance Statements for SMIv2, McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.

IETF RFC 2863, The Interfaces Group MIB using SMIv2, McCloghrie, K. and Kastenholz, F., June 2000.

~~IETF RFC 2869, RADIUS Extensions, Rigney, C., Willats, W., and Calhoun, P., June 2000.~~

IETF RFC 3394, Advanced Encryption Standard (AES) Key Wrap Algorithm, J. Schaad, R. Housley, September 2002.

~~IETF RFC 3410, Introduction and Applicability Statements for Internet Standard Management Framework, J. Case, R. Mundy, D. Partain, B. Stewart, December 2002.~~

~~IETF RFC 3579, RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP), Aboba, B., Calhoun, P., September 2003.~~

~~IETF RFC 3580, IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Guidelines, Congdon, P., Aboba, B., Smith, A., Zorn, G., Roese, J., September 2003.~~

IETF RFC 3629, STD 63, UTF-8, a transformation format of ISO 10646, Yergeau, F., November 2003.

~~IETF RFC 4017, Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs, Stanley, D., Walker, J., Aboba, B., March 2005.~~

IETF RFC 4346, The Transport Layer Security (TLS) Protocol Version 1.1, Diercks, T., Rescorla, E., April 2006.

IETF RFC 4493, THE AES-CMAC Algorithm, Song, J.H., Lee, J., Iwata, T., June 2006.

~~IETF RFC 4675, RADIUS Attributes for Virtual LAN and Priority Support, Congdon, P., Sanchez, M., Aboba, B., September 2006.~~

IETF RFC 5216, The EAP-TLS Authentication Protocol, Simon, D., Aboba, B., Hurst, R., March 2008.

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

IETF RFC 5247, Extensible Authentication Protocol (EAP) Key Management Framework, Aboba, B., Simon, D., Eronen, P., October 2007.

~~FIPS Publication 197, The Advanced Encryption Standard (AES), U.S. DoC/NIST, November 26, 2001.~~

ISO/IEC 18033-3: 2010, Information technology—Security techniques—Encryption algorithms—Part 3:Block ciphers.

~~NIST Federal Information Processing Standard 140-2, Security Requirements for Cryptographic Modules[6], 3 December 2002.~~

~~NIST Special Publication 800-38B, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, Morris Dworkin, May 2005.[7]~~

~~NIST Special Publication 800-90, Recommendation for Random Number Generation Using Deterministic Random Bit Generators, E. Barker, J. Kelsey, revised March 2007.~~

NIST Special Publication 800-108, Recommendation for Key Derivation Using Pseudorandom Functions, Lily Chen, November 2008.

[6] ~~National Institute of Standards and Technology, FIPS 140-2 is available at http://www.nist.gov/cmvp.~~

[7] NIST Special Publications (800 Series) are available at http://csrc.nist.gov/publications/PubsSPs.html.

3. Definitions

Change the definition of packet number as follows:

packet number (PN): A monotonically increasing value ~~used to uniquely identify a MACsec frame in the sequence of frames transmitted using an SA~~ that is guaranteed unique for each MACsec frame transmitted using a given SAK.

Insert the following definition(s), in the appropriate collating order:

extended packet number (XPN): A 64-bit packet number (PN) specified in IEEE Std 802.1AE.

Salt: A 96-bit secret value communicated by key agreement protocol for use by the protection and verification operations of the IEEE Std 802.1AE GCM-AES-XPN Cipher Suites.

Short Secure Channel Identifier (SSCI): A 32-bit value that is unique for each SCI within the context of all SecYs using a given SAK.

NOTE—IEEE Std 802.1AEbw-2013 specifies the calculation of SSCI and Salt values used by the IEEE Std 802.1AE GCM-AES-XPN Cipher Suites from other MKA values.

4. Abbreviations and acronyms

Insert the following abbreviation(s), in the appropriate collating sequence:

SSCI	Short SCI
XPN	Extended Packet Number

Delete the following abbreviation:

FDDI	Fiber Distributed Data Interface

5. Conformance

5.11 MKA options

Insert new subclause 5.11.4 as follows:

5.11.4 In-service upgrades

A PAE that supports in-service upgrades shall be capable of

a) Suspending MKA operation as specified in 9.18.
b) Communicating the values of the most significant 32 bits of the Lowest Acceptable PN for the Latest Key and the Old Key when any XPN capable Cipher Suite is being used, as specified in 9.18.5.

NOTE—Selection and use of Extended Packet Numbering depends on the implementation of an XPN capable Cipher Suite by each SecY participating in a CA. See IEEE Std 802.1AE as amended by IEEE Std 802.1AEbw-2013.

A PAE that supports in-service upgrades may use additional protocol(s), outside the scope of this specification, to coordinate in-service upgrades as specified in 9.18.6.

6. Principles of port-based network access control operation

6.2 Key hierarchy

Change the first paragraph of 6.2 as follows:

The root key in the MACsec Key Agreement key hierarchy is the Connectivity Association Key (CAK), and is identified by a CAK Name (CKN). MKA (Clause 9) does not use the CAK directly but derives two further keys from the CAK using the AES cipher (~~FIPS Publication 197~~ISO/IEC 18033-3) in CMAC mode (9.3). These are the ICV Key (ICK) used to verify the integrity of MPDUs and to prove that the transmitter of the MKPDU possesses the CAK, and the Key Encrypting Key (KEK) used by Key Server, elected by MKA, to transport a succession of SAKs, for use by MACsec, to the other member(s) of a CA. See Figure 6-3.

Change 6.2.4 as follows:

6.2.4 Algorithm agility

To accommodate future developments in cryptography, MKA provides an explicit Algorithm Agility parameter (9.3.3, 11.11.2, Figure 11-8). The Algorithm Agility parameter identifies the following:

a) How the ICV is derived from the CAK and the data of a given MKPDU; and
b) How a CAK is derived from the parameters available to the participants in an EAP exchange.

Knowledge of item a) is required for MKA's use of CAKs in general, while both item a) and item b) are required for the validation of MKPDUs that are protected using the results of an EAP exchange.~~In other words,~~ MKA instances that transmit MKPDUs with different values of the Algorithm Agility parameter could use a different KDF (see 6.2.1) to derive the ICK (9.3.3), could specify the computation of the ICV in a different way, and could specify a different way for the participants in an EAP exchange to agree on a CAK (6.2.2). The flexibility provided allows a wide range of future challenges to be addressed, but the need for substantial analysis of any proposed alternate to the provisions of this standard in these areas means that no claim of conformance is facilitated currently for any such alternate. A future revision of this standard could provide such a claim, if needed to address cryptographic developments.

MKA's Algorithm Agility parameter does not identify the KDF used to derive the KEK, or the Key Wrap (see 6.2). ~~That identification is provided~~ They are identified by the MKA parameter set type used to encode the Key Wrap. MKA parameter sets 4 and 5 (Figure 11-11, Figure 11-12, Figure 11-13) use a particular KDF (6.2.1, 9.3.3) and Key Wrap (AES Key Wrap, 9.8.2, 9.12.1) by definition. A different Key Wrap, or a Key Wrap using a differently derived KEK, could be introduced by defining a further parameter set type while still allowing MKA participants to communicate by using MKPDUs with a familiar Algorithm Agility parameter value—thus allowing negotiation or fallback to known parameter set types.

7. Port-based network access control applications

7.7.2 System configuration and operation

Change the note to Figure 7-16 as follows:

NOTE—Figure 7-16 is based on Figure 15-1 of IEEE Std 802.1Q ~~(as amended by IEEE Std 802.1ad)~~.

Change the last sentence of the second paragraph of 7.7.2 as follows:

The interface stack shown includes the service access priority selection function described in 6.9 of IEEE Std 802.1Q ~~(as amended by IEEE Std 802.1ad)~~.

9. MACsec Key Agreement protocol (MKA)

Insert the following item(s) at the end of the list introduced by "This clause specifies …"

o) Management of the KaY and MKA (9.16).

p) Temporary suspension of MKA operation to facilitate in-service control plane software upgrades without disrupting existing secure connectivity (9.18).

Change the dashed list in the introductory text of Clause 9 as follows:

The following terms are used to identify roles within the protocol or protocol scenarios:

— **participant:** The personification of a single KaY's participation in a given MKA instance (i.e., transmitting and receiving MKPDUs protected by keys derived from a single given CAK and identified by a given CKN), operating with positive intent, and obeying the protocol.

— **actor:** The participant under discussion, usually in the KaY being described.

— **partners:** Other participants in the same MKA instance, and attached to the same LAN, as the actor.

— **successful actor:** An actor that has one or more live partners and is participating in an MKA instance that has elected a Key Server.

— **principal actor:** The successful actor selected by a KaY to control its associated PAC or SecY. ~~that is participating in the MKA instance with the highest priority Key Server.~~

— **member**: ~~A participant using MKA to establish Controlled Port connectivity with other participants.~~ The personification of a single KaY's participation in all MKA instances and use of other controls that determine the connectivity provided by its associated Controlled Port. A system is described as being member of a CA if it includes a Controlled Port providing connectivity using that CA.

9.1 Protocol design requirements

Insert an additional note after bullet c) as follows:

c) Following any period of 8 seconds during which all frames transmitted by each of a set of participants are delivered, once and without misordering, to each of the other participants, and the load imposed by frames received from an attacker does not exceed the resources of any participant, MKA will provide the keys and information required by each of its participants' clients, irrespective of the prior state of each participant or frames buffered by the LAN.

NOTE 1—The requirement is that of correctness: the protocol convergence time is bounded; rather than a performance goal. The figure of 8 seconds arises from the possibility that prior participants are being timed out (over a period of MKA Life Time, 6 seconds, see Table 9-3) just as a new participant joins, plus MKA Hello Time (2 seconds) to ensure that all participants have subsequently transmitted.

NOTE 2—If MKA operation is suspended (9.18), and the participants do not already possess the necessary keys and information, convergence will be delayed until MKA operation resumes.

Change the note following bullet k) as follows:

k) It allows its participants to ensure that the data frames protected by MACsec are not being delayed by more than 2 seconds.

NOTE ~~2~~3—Delay protection guards against an attack on the configuration protocols that MACsec is designed to protect by alternately delaying and delivering their PDUs. Delay protection does not operate if and when MKA operation is suspended (9.18).

Insert the following new text after the existing text of 9.1:

When the option to support MACsec Cipher Suites that use Extended Packet Numbering is implemented, MKA meets the following requirements:

m) Recovery, if lost, of the 32 most significant bits of the XPN, in support of the protocol correctness requirement c) above.

9.3.3 Derived keys

Change the text of 9.3.3, moving the fourth paragraph to become the second, and reversing the order of the present second and third paragraphs, so the entire text of the subclause (showing changes made to each paragraph) is as follows:

Each of the keys used by MKA is derived from the CAK ~~using the AES Cipher in CMAC mode (NIST Special Publication 800-38B)~~. The CAK is not used directly. The derived keys are tied to the identity of the CAK, and thus restricted to use with that particular CAK. ~~The KDF specified in 6.2.1 is used to derive these keys. The hexadecimal representations of each of the text strings used with this KDF is given in H.~~

To accommodate future developments in cryptography, each MKPDU conveys an Algorithm Agility parameter that identifies how the ~~KEK and~~ ICK ~~are~~ is derived from the CAK~~,~~ and how it is ~~these keys are~~ used. Each Algorithm Agility parameter value comprises four octets, the first three being those of an OUI (Organizationally Unique Identifier) allocated by the IEEE Registration Authority, and the fourth allocated by the organization to which that OUI has been allocated. The derivation of the ~~KEK and~~ ICK, and the use ~~and size~~ of ~~these~~ this key~~s~~ as specified in this standard, is identified by the value specified in Table 9-1.

The ICK is derived from the CAK using the KDF specified in 6.2.1. This KDF uses the AES Cipher in CMAC mode (IETF RFC 4493). The ICK is derived from the CAK using the following transform:

ICK = KDF(Key, Label, Keyid, ICKLength)

where

Key = CAK

Label = “IEEE8021 ICK”

Keyid = the first 16 octets of the CKN, with null octets appended to pad to 16 octets if necessary

ICKLength = two octets representing an integer value (128 for a 128 bit ICK, 256 for a 256 bit ICK) with the most significant octet first

~~In each case t~~The Label is a UTF-8 string, without a null or other termination, exactly 12 bytes in length (the quotes shown do not form part of the string and exactly one space separates ‘8021’ ~~and ‘KEK’, and ‘8021’~~ and ‘ICK’). The length of the Label is chosen to make the input to the PRF within the KDF exactly 32 bytes. The hexadecimal representations of each of the text strings used with this KDF and test vectors are given in Annex H.

To accommodate future developments in cryptography, each of the MKA parameter set types used by MKA to communicate a wrapped key identifies the Key Wrap Algorithm and KEK derivation (6.2.4, Figure 11-11, Figure 11-12, Figure 11-13) used. The KEK used by all parameter set types currently specified in this standard is derived from the CAK using the KDF specified in 6.2.1. This KDF uses the AES Cipher in CMAC mode (IETF RFC 4493). The KEK is derived from the CAK using the following transform:

KEK = KDF(Key, Label, Keyid, KEKLength)

where

Key = CAK
Label = “IEEE8021 KEK”
Keyid = the first 16 octets of the CKN, with null octets appended to pad to 16 octets if necessary
KEKLength = two octets representing an integer value (128 for a 128 bit KEK, 256 for a 256 bit KEK) with the most significant octet first

~~In each case t~~The Label is a UTF-8 string, without a null or other termination, exactly 12 bytes in length (the quotes shown do not form part of the string and exactly one space separates ‘8021’ and ‘KEK’~~, and ‘8021’ and ‘ICK’~~). The length of the Label is chosen to make the input to the PRF within the KDF exactly 32 bytes.

When SAKs and CAKs are distributed they are protected by the KEK, using an AES Key Wrap as defined in 9.8.2 and 9.12.1. ~~The number of bits in each SAK, and its use, is not tied to the value of MKA’s Algorithm Agility parameter but is identified by the value of the MACsec Cipher Suite parameter.~~

SAKs should be derived from the CAK as specified in 9.8.1, but may also be generated directly by the Key Server’s strong random number generator (RNG, 9.2.1). Distributed CAKs, when used, shall be random values generated by the MKA Key Server RNG. Each distributed CAK is distinct from previously distributed CAKs, so that an MKA participant or attacker holding only the current CAK cannot determine a previously distributed CAK. This allows implementation of a policy of perfect forward security, with a fresh CAK being distributed when each participant joins a CA, so that participant cannot decrypt wrapped keys from previously transmitted MKA frames.

NOTE—MKA does not require fresh CAK distribution when a new participant joins a CA, as that would prolong the process of joining.

Change Table 9-1 (correcting it to match the approval date and designation of IEEE Std 802.1X-2010, as identified in the text of 9.3.3 as “this standard”) as follows:

Table 9-1—MKA Algorithm Agility parameter values

Parameter value	Specification
00-80-C2-01	IEEE Std 802.1X-20~~09~~10

9.4.3 Determining liveness

Change the text and notes of 9.4.3 as follows:

A participant proves liveness to each of its peers by including their MI, together with an acceptably recent MN, in an MKPDU with the participant’s own MI and MN.

To avoid a new participant having to respond to each MKPDU from each partner as it is received, or trying to delay its reply until it is likely that MI.MN tuples have been received from all potential partners, each participant maintains and advertises both a Live Peer~~s~~ List and a Potential Peer~~s~~ List. The Live Peer~~s~~ List includes all the peers that have included the participant’s MI and a recent MN in a recent MKPDU, and the Potential Peer~~s~~ List includes all the other peers that have transmitted an MKPDU that has been directly received by the participant or that were included in the Live Peer~~s~~ List of an MKPDU transmitted by a peer that has proved liveness. Peers are removed from each list when an interval of between MKA Life Time (see Table 9-3) and MKA Life Time plus MKA Hello Time has elapsed since the participant’s recent MN (see above) was transmitted. This time is sufficient to ensure that two or more MKPDUs will have been lost or delayed prior to the incorrect removal of a live peer.

NOTE 1—The specified use of the Live and Potential Peer Lists thus permits rapid removal of participants that are no longer active or attached to the LAN while reducing the number of MKPDUs transmitted during group formation. For example, a new participant will be admitted to an established group after receiving, then transmitting, one MKPDU.

NOTE 2—A suspended participant (9.18) will be removed from the Live and Potential Peer Lists as described, but its associated SecY will still be able to transmit and receive secure frames until other CA members adopt a new SAK.

Insert a new subclause 9.4.6 as follows:

9.4.6 Active and passive participants

A participant can be active, transmitting periodic MKPDUs, or passive. A passive participant will become active for a period of MKA Lifetime following the receipt of an MKPDU from a feasible partner, i.e., provided that either the receiving participant or the partner is prepared to act as a Key Server. Whether a participant is to be active when first created, and whether it is to remain active in the absence of feasible partners depends on the port-based network access control application. The creation of passive participants supports systems that have many potential peers, with only one or a few likely to be connected at a time. Participants that are always active are desirable where connectivity is provided by media that do not reliably signal loss and resumption of connectivity, as can be the case for infrastructure links supported by virtual media. If all the participants in a potential CA can be passive, and an extended and undetected network outage occurs, it is possible that the potential CA members will fail to transmit MKPDUs, resulting in a permanent lack of connectivity.

NOTE 1—The condition of having recently received an MKPDU from a feasible partner can be determined by inspecting the participant’s Live Peer List and Potential Peer List.

NOTE 2—The model of Logon Process operation encompasses participant creation, deletion, and control of active or passive participation (12.2, 12.5.2). The CAK cache provides management (see activate in 9.16).

An active participant will remain active, even in the absence of received MKPDUs, while a suspension is in progress (provided that the participant is not itself suspended). A suspended participant will resume active operation if and only if

a) Its CAK was previously cached, and
b) The management controls associated with that cached CAK specify that it is to be active on resumption (9.16).

9.5 Key server election

Change the list in 9.5 as follows:

The participants in a given MKA instance agree on a Key Server, responsible for the following:

a) Deciding on the use of MACsec (9.6)
b) Cipher suite selection (9.7)
c) SAK generation and distribution (9.8)
d) SA assignment (9.10.1)
e) Identifying the CA when two or more CAs merge (9.15)
f) CA formation and group CAK distribution (9.15)
g) Initiating, continuing, and terminating MKA suspension (9.18)

If the CAK is a pairwise CAK derived directly from EAP (see 6.2.2), the MKA participant for the PAE that was the EAP Authenticator will be the Key Server, and will not accept information [a) through ~~f~~g) above] from any other participant that attempts to act as the Key Server for that MKA instance.

Change the fourth paragraph of 9.5 as follows:

If a KaY participates in multiple MKA instances so that there are several actors—one per instance—for a given port, then only the actor selected as the principal actor (12.1) will (if elected Key Server for its CAK) distribute SAKs ~~and perform other Key Server functions~~. ~~An actor that has been elected a Key Server by its peers but is not the principal actor for its KaY (12.1) will, by not choosing and distributing the parameters necessary for communication to its peers, ensure that a single principal actor is chosen for all the MKA Entities that participate in overlapping CAK, CKN distribution on a LAN, and that a single unambiguous MACsec CA can be formed on that LAN.~~ Since the factors that cause a principal actor to be selected from its peers are the same for different CAK, CKN tuples with the same distribution, replacement of a principal actor by its successor will occur, whenever possible, without a change of Key Server. This succession plan ensures that the SAKs distributed using one CAK, CKN tuple can be followed immediately by SAKs distributed by a successor CAK, CKN without any loss of MACsec connectivity. To minimize the chance of a CA member that possesses both CAKs temporarily losing connectivity, a Key Server should not distribute an SAK using the new CAK until MKA Life Time (Table 9-3) has elapsed after it has started participating with that CAK, and should not delete the participant for the prior CAK until MKA Life Time has elapsed after that new SAK is first distributed. The CP state machine (Figure 12-2) ensures that a new SAK is not distributed until the Key Server is receiving and transmitting using a single SAK.

Delete the current note and insert two new notes as follows:

~~NOTE—Choice of the principal actor by each KaY to receive information from, or to act as, the Key Server ensures that two PAEs will choose the same Key Server even if they both comprise EAP Authenticator and Supplicant functionality, and each has authenticated as the Authenticator. The KaY with the highest priority will be the Key Server.~~

NOTE 1—If two PAEs are each capable of acting as both an EAP Authenticator and an EAP Supplicant, their interaction can result in two instances of successful mutual authentication with each acting as the Authenticator in one. Thus two MKA instances can be created, and each PAE’s KaY will be elected Key Server for one instance. However, each KaY will select, as its principal actor, its participant in the MKA instance with the highest priority Key Server. That Key Server can then use that MKA instance to distribute SAKs (or select unauthenticated connectivity, see 12.3), and the participants in the other MKA instance can be deleted.

NOTE 2—A number of KaYs participating in multiple MKA instances will still succeed in configuring a single CA even if each participates in a different subset of those instances, provided that the highest priority Key Server in each subset itself participates in an MKA instance with a higher priority Key Server or is the highest priority Key Server. For example, assume KaYs A, B, C, D (say) in decreasing priority order, with two MKA instances with participants {A, B, C} and {B, D} respectively. Although B will be elected Key Server for the {B, D} instance, it will not distribute SAKs to D, as B’s principal actor will be in {A, B, C}. Thus a single CA will be created, including the ports associated with A, B, and C, but excluding D's. However, scenarios of this type are most likely to result from errors in manual key distribution. They can give rise to temporary interruptions or unwanted connectivity, particularly where in-service upgrades are to be performed and need to be eliminated as part of managing upgrades (9.18.6).

9.6.1 MKPDU application data

Change 9.6.1 as follows:

Each MKA participant encodes the following information in every MKPDU transmitted:

a) MACsec capability, indicating whether MACsec is implemented, and if so whether the implementation provides integrity protection only, integrity and integrity with confidentiality, or integrity and integrity with confidentiality with a selectable confidentiality offset of 0, 30, or 50 octets (see Table 11-6, IEEE Std 802.1AE-2006).

   NOTE—IEEE Std 802.1AE-2006 introduced the confidentiality offset to facilitate early MACsec deployment on existing systems that needed to store received frames before applying MACsec processing and that needed to examine the initial octets of received frames to decide where to store those frames. The XPN Cipher Suites standardized in IEEE Std 802.1AEbw-2013 do not support confidentiality offsets of other than 0.

b) MACsec desired, a flag, set if the participant desires the use of MACsec to protect frames.

9.7.1 MKPDU application data

Change 9.7.1 as follows:

A participant that believes itself to be the Key Server and its KaY's principal actor encodes the following information with each MACsec SAK that it distributes, unless the mandatory Default Cipher Suite GCM-AES-128 is to be used:

a) MACsec Cipher Suite, the Cipher Suite reference number.

The following information is also distributed with each MACsec SAK:

b) Confidentiality Offset, indicating whether confidentiality is to be provided, and whether an offset of 0, 30, or 50 octets is used (see IEEE Std 802.1AE-2006).

If a receiving MKA participant does not implement the referenced Cipher Suite with the selected confidentiality offset, the distributed SAK will not be installed (12.4). An MKA participant should advertise any Cipher Suites implemented in addition to the Default Cipher Suite by including an Announcement parameter set (11.11.1 Figure 11-15) with a MACsec Cipher Suites TLV (11.12.3) in each MKPDU transmitted.

NOTE—~~This standard does not currently provide any way for MKA to negotiate the use of an alternative cipher suite or a confidentiality offset. While the format of an MKPDU (Clause 11) can accommodate the definition of additional parameter sets, this standard deliberately does not provide a way to add non-standard parameters.~~ The XPN Cipher Suites standardized in IEEE Std 802.1AEbw-2013 do not support confidentiality offsets of other than 0.

9.8 SAK generation, distribution, and selection

Change the text of the third and fourth paragraphs of 9.8, as follows:

The Key Server observes the Lowest Acceptable PN (LLPN) for the Latest Key in use, as transmitted by each CA member, and shall distribute~~s~~ a fresh SAK whenever a participant advertises a Latest Key Identifier (LKI) that matches the KI of the key currently being distributed and an LLPN that equals or exceeds the constant PendingPNExhaustion. PendingPNExhaustion is 0xC000 0000 for 32-bit PNs and 0xC000 0000 0000 0000 for 64-bit PNs. Subject to conditions [a) through c), below] that limit the frequency of SAK changes, postponing their generation and distribution until CA membership is likely to be stable, the ~~The~~ Key Server shall also distribute a fresh SAK whenever a member is added to the live membership of CA (as perceived by the Key Server—with each MI, not the associated SCI, representing each member), and can distribute a fresh SAK when a member is removed from the live membership. A fresh SAK is not distributed until:

a) The Key Server's Live Peer List contains at least one peer, and

b) An MKA suspension is not in progress, i.e., the Key Server either does not support suspension (5.11.4), or the Key Server's suspendedWhile timer is zero (9.18), and

c) Either

   1) MKA Life Time (Table 9-3) has elapsed since the prior SAK was first distributed, or

   2) The Key Server's Potential Peer List is empty.

~~A fresh SAK is not generated until the Key Server's Live Peer List contains at least one peer, and~~

a) ~~MKA Life Time (Table 9-3) has elapsed since the prior SAK was first distributed, or~~

b) ~~The Key Server's Potential Peer List is empty.~~

Once a Key Server has generated an SAK, it shall be distributed in each MKPDU transmitted by its principal actor until all live peers that can use the selected Cipher Suite and Cipher Suite capability (9.6.1) report having installed the SAK for receive or until a change in the live membership of the CA requires the generation of a fresh SAK.

9.10.1 MKPDU application data

Change bullet c) of 9.10.1, as follows:

c) LPN, Lowest Acceptable PN (least significant 32 bits for XPN Cipher Suites)

A fixed format encoding is supported by an 'In Service' flag, indicating that the fields for the respective SA are being used. For convenience, these fields can be identified by the names and acronyms Latest In Service/Old In Service (LIS/OIS), Latest AN, Old AN (LAN, OAN), Latest Key Identifier/Old Key Identifier (LKI/OKI), Lowest Acceptable PN for the Latest Key/Lowest Acceptable PN for the Old Key (LLPN/OLPN), Latest Receiving/Old Receiving (LRX/ORX), Latest Transmitting/Old Transmitting (LTX/OTX).

Change the note in 9.10.1 and insert an additional note as follows:

NOTE 1—The Latest and Old SAKs were not necessarily distributed by the same Key Server, or by the current Key Server. Both can be receiving at the same time, to enable transition from one SAK to the next without frame loss, although only one will be transmitting at any instant.

NOTE 2—When an XPN Cipher Suite is used the most significant 32 bits of the Lowest Acceptable PNs for both ANs is encoded as specified in 9.18.7.

9.15 MKA participant timer values

Change Table 9-3, inserting an additional row as follows:

Table 9-3—MKA Participant timer values

Timer use	Timeout (parameter)	Timeout (seconds)
Per participant periodic transmission, initialized on each transmission, transmission on expiry (9.4).	MKA Hello Time or MKA Bounded Hello Time	2.0 0.5
Per peer lifetime, initialized when adding to or refreshing the Potential Peer~~s~~ List or Live Peer~~s~~ List, expiry cause removal from the list (9.4.3).	MKA Life Time	6.0
Participant lifetime, initialized when participant created or following receipt of an MKPDU, expiry causes participant to be deleted (9.14).		
Delay after last distributing an SAK, before the Key Server will distribute a fresh SAK following a change in the Live Peer List while the Potential Peer List is still not empty.		
Maximum suspendFor value. The maximum suspendedWhile value is MKA Life Time longer.	MKA Suspension Limit	120.0

9.16 MKA management

Change the introductory paragraph of 9.16 as follows:

The PAE management process controls and monitors the operation of the KaY and MKA participants, providing access for network management through the LMI. The following variables (see also 12.2) can be used to manage the operation of the KaY ~~as~~ for a given port (as identified by its portNumber, see 12.9):

Change the definitions of the variables 'authenticated', 'formGroup', and 'newGroup' in the first dashed list in 9.16 as follows:

— authenticated: Set if the principal actor~~, i.e. the participant that has the highest priority Key Server and one or more live peers,~~ has determined that Controlled Port communication should proceed without MACsec.

— formGroup: Set if the KaY will attempt to use point-to-point CAKs to distribute a Group CAK, if it~~s principal actor~~ is the Key Server for the MKA instances for all the point-to-point CAKs.

— newGroup: Set by management if a new Group CAK is to be distributed, if the KaY ~~principal actor~~ is the Key Server for the MKA instances for all the point-to-point CAKs. Cleared by the KaY when distribution is complete.

Add the following variables to the first dashed list in 9.16, after rxAN:

— suspendFor: Set by management to a non-zero number of seconds between 1 and MKA Suspension Limit to initiate a suspension (9.18) of that duration (if the KaY's principal actor is the Key Server) or to request a suspension (otherwise).

— suspendOnRequest: Set by management to allow the KaY's principal actor to initiate a suspension if it is the Key Server and another participant has requested a suspension.

- suspendedWhile: Read by management to determine if a suspension is in progress and (when available) to discover the remaining duration of that suspension.

Change the bullet item beginning "— activate ..." as follows:

- activate {Default, Disabled, OnOperUp, Always}: Controls when the participant is activated. Cached entries created by the KaY as part of normal operation, without explicit management, have the value Default, and are activated according to the implementation dependent policies of the KaY (see 9.15). This variable can be set to any of its values by management. Disabled allows the cache entry to be retained, but disabled for an indefinite period. OnOperUp causes the participant to be activated when the PAE's Uncontrolled Port ~~port (and therefore when the SecY or PAC's Common Port~~ becomes MAC_Operational~~)~~ and when the PAE resumes following suspension (9.18). Always causes the participant to remain active all the time, even in the continued absence of partners. If the value is changed to Disabled~~or OnOperUp~~, the participant ceases operation (is deleted) immediately~~ and receipt of MKPDUs with a matching CKN during a subsequent period of twice MKA Life Time will not cause the participant to become active once more~~.

Insert new subclauses 9.18 and 9.19 as follows:

9.18 In-service upgrades

The control plane software of a system that is a member of a CA can be upgraded, temporarily suspending MKA operation, without interrupting the secure data connectivity provided by the CA, if the system and each of its peers in the CA support in-service upgrades (5.11.4). An SAK that is already in use will continue to be used provided that each of these peers does not conclude that it is the CA's only active member, and provided that a fresh SAK is not distributed by a recognized Key Server. If each member's suspendedWhile timer is not zero these conditions will be met (12.2, 12.4, 12.5).

NOTE—A KaY's suspendedWhile timer takes a non-zero value to indicate a suspension, a period during which one or more of its principal actor's partners has ceased (or can be expected to cease) transmitting MKPDUs.

This standard specifies procedures that allow a Key Server to control the initiation, proposed duration, and possible early termination of any MKA suspension by communicating the current value of the suspendedWhile timer in MKPDUs. A Key Server that supports in-service upgrades always includes the value in each MKPDU transmitted. A suspension is terminated by expiry of a participant's suspendedWhile timer, by the Key Server communicating a zero value, or by a Key Server that does not support in-service upgrades omitting the value from transmitted MKPDUs.

9.18.1 Initiating suspension

A participant other than the Key Server can request a suspension by transmitting a (non-zero) suspendFor value in an MKPDU. The participant can maintain the request for an indefinite period by repeating the value in each MKPDU transmitted before it suspends MKA operation. If the request is no longer relevant, perhaps because the participant has upgraded its control plane software without waiting for a suspension, the participant can drop the request by transmitting MKPDUs with a suspendFor value of zero.

The Key Server records, for each of the participants on its Live Peer List, the lesser of the values of the suspendFor parameter in the last MKPDU received and the MKA Suspension Limit specified in Table 9-3.

A Key Server should not initiate a suspension until it has started transmitting using the last SAK it has distributed and is no longer receiving using any prior SAK. This requirement is equivalent to stating that the CP state machine should be in state RETIRE having completed the actions on entry to that state (see Figure 12-2). If these conditions are satisfied, and the value of the Key Server's own suspendFor parameter is non-zero or the Key Server's policy control suspendOnRequest is True and one of the received suspendFor parameters is non-zero, the Key Server will initiate a suspension. The Key Server can apply additional policy controls on the setting of its suspendFor and suspendOnRequest parameters to limit when a suspension can occur, or to limit the frequency of suspensions. Any participant can also apply its own policy controls, limiting (for example) how long it is prepared to wait for the Key Server to initiate a suspension of adequate duration.

To initiate a suspension the Key Server sets the value of its suspendedWhile timer to the greatest of the applicable suspendFor values. Subsequently suspendedWhile is decremented once per second, and then set to the greater of its new value and the greatest applicable suspendFor value. The value of suspendedWhile (decrementing over time) is transmitted in all MKPDUs for all MKA instances that have elected it Key Server. These transmissions should persist for at least MKA Life Time before the Key Server suspends its own operation (if desired).

A participant, other than the Key Server, that wishes to suspend its own operation of MKA includes a non-zero suspendFor value in all MKPDUs transmitted and should not suspend its operation of MKA until it receives an MKPDU from the Key Server with a non-zero suspendedWhile value that is greater than or equal to the value of its suspendFor parameter.

NOTE—Details of MKA operation are likely just a part of the concerns to be addressed when performing a system software upgrade, and the overall considerations will not necessarily allow the KaY to delay suspension to facilitate transmission and reception of MKPDUs as recommended. The probability that the transmission of a single MKPDU by the participant requesting suspension will be successfully received and acted on by the Key Server might be acceptable, as might the probability of reception of a single MKPDU transmitted by the Key Server.

If the requesting participant does not suspend its own operation of MKA, but continues to transmit a non-zero suspendFor value in subsequent MKPDUs, then the Key Server's suspendedWhile value will continue to be at least that of the requested value—provided that the Key Server does not suspend its own operation and its suspendOnRequest parameter remains True. The MKA Suspension Limit specified in Table 9-3 limits the time for which a participant will maintain existing connectivity after all partners requesting suspension have been removed from its Live Peer List, on the assumption that these partners have been suspended and will resume operation.

9.18.2 Suspending

A CA member should not suspend its own operation, unless it is

a) Receiving and transmitting using a single SAK, or
b) Constrained by policy controls that place limits on the time that it is prepared to wait to suspend.

9.18.3 Suspended members

A CA member that suspends MKA operation while using MACsec to secure connectivity continues to generate, transmit, receive, and verify secure data frames as specified by IEEE Std 802.1AE, maintaining the receive and transmit SAs current at the time of the suspension. In effect, the CP state machine remains in state RETIRE.

A suspended member is not required to retain any record of its MI, or of its Live Peer List or Potential Peer List, for any of its participants. It requires a record of any SAK(s) used by transmit and receive SAs only in so far is required by the operation of those SAs, and shall not redistribute or otherwise share the SAK(s) with other participants while suspended or at any later time.

A suspended member does require access, when operation is resumed, to the following information required by the operation of each SA in use:

a) The SCI

b) The AN

A suspended member is not required to retain the KI of any SAK in use, but can report a zero value (after resuming operation) when responding to management requests and completing KI fields in MKPDUs. The rules for SAK generation (9.8) ensure that a fresh SAK will be distributed after a suspended member resumes operation with a new MI.

A member can suspend MKA operation because part of its system's functionality will not be available during an in-service upgrade. The member might not be able to transmit and receive frames because software associated with a particular interface module is to be upgraded, for example. The suspended member can retain its MI provided that it continues to operate MKA, with the exception that any MKPDUs that might be generated for transmission or that are received can be discarded. The normal operation of MKA will result in the removal of partners from the Live Peer List and Potential Peer List (9.4.3) if MKPDU loss persists.

9.18.4 Resuming operation

When a CA member resumes, it sets its own suspendedWhile timer value to Max Suspension Limit or to some lower policy determined limit. It also sets its suspendFor parameter value to zero once it has determined that the upgrade has completed successfully (see 9.18.6).

NOTE 1—The Max Suspension Limit is specified in Table 9-3.

The CA member might have suspended only because it was unable (for the duration of the suspension) to receive or transmit MKPDUs or install fresh SAKs. If it has continued to operate MKA while suspended, its MI, Live Peer List, and Potential Peer List will be retained when it resumes. Otherwise, i.e., if MKA operation ceased, it will select a fresh MI and its Live Peer List and Potential Peer List will be initialized and empty on resumption.

If the Key Server that initiated the suspension suspends itself, then it is possible that a participant for another CA member (that might or might not have suspended itself) will be elected Key Server before the initiating Key Server resumes. If the newly elected Key Server's suspendedWhile timer is running, then it will distribute the (decrementing) value of that timer and thus prolong the suspension. This provision means that it is unnecessary for a resuming member to remember whether it was or was not the Key Server prior to being suspended. If the initiating Key Server resumes operation before the timer expires it will once more assume responsibility for monitoring the suspension.

The current Key Server shall terminate the suspension before the value of its suspendedWhile timer reaches zero by resetting that value (included in all transmitted MKPDUs) to zero under the following conditions:

a) Either
   1) For every active receive SA it has a live peer with an SCI that matches that of the receive SA;
   or
   2) It has a live peer with an SCI that does not match any existing receive SA;

   and
b) It has recorded a value of zero for the suspendFor parameter received from each of these live peers.

To determine, for the purpose of test 1), that a receive SA is active the Key Server monitors the value of the InPktsOK management counter (see IEEE Std 802.1AE–2006 10.6.5 and Table 13-6) for the SA. The receive SA is not active if the counter has not been incremented for MKA Life Time. Test 2) detects the addition of a new CA member.

NOTE 2—For the common case of a point-to-point CA, these conditions simplify to either having a live peer communicating a zero suspendFor value or not having received any secured frames for MKA Life Time.

NOTE 3—While the receipt of validated secured data frames does not ensure that communication is with a live peer in current possession of a shared SAK, the lack of any such reception for a period of MKA Life Time (6 seconds) is a strong practical indication of the absence of such a peer.

9.18.5 XPN support

If a receiver loses more than 2^{30} consecutive frames when an XPN capable Cipher Suite is being used, the 32 most significant bits of the PN of the next frame to be received might be recovered incorrectly. If more than 2^{32} frames are lost, these most significant bits will be recovered incorrectly. All subsequent secured frames will fail validation and be discarded, unless some means other than the receipt of secured data frames is used to determine their value.

When a suspension is in progress, the risk of losing a large number of frames is increased as the operation of supervisory protocols that would otherwise detect temporary loss of connectivity might also be suspended. At 100 Gb/s 2^{30} minimum sized secured frames can be transmitted in 10 seconds, well within the potential duration of a suspension. To ensure that the most significant bits are recovered when all CA members resume MKA operation, the most significant 32 bits of the Lowest Acceptable PN for the Latest Key and the Old Key are communicated when in-service upgrades are supported and any XPN capable Cipher Suite is being used.

9.18.6 Managing in-service upgrades

Careful planning is required when upgrading systems that compose a network if the network is to remain in operation, and if the costs of recovering from a failed upgrade are considerable. Best practices include the following:

a) Ensuring that the network manager has an up to date record of all the systems in the network, and of the network configuration.
b) Ensuring that the network manager has an up to date and independently backed up record of the software, software revision levels, and configuration parameters currently used by the systems to be upgraded, by their neighbors, and by their other peers in the network.
c) Off-line verification and testing for compatibility between the proposed new software and configuration parameters and the existing software and configuration of neighbors and other peers.
d) Retention of the existing software and configuration parameters by the systems being upgraded until successful operation with the new software and parameters is confirmed.
e) Use of a ‘dead man’ timer by the system to be upgraded, so that the system will automatically revert to the prior software and configuration if satisfactory management communication cannot be established with the network operations center after the upgrade.
f) Off-line verification and testing of any such fallback mechanism.
g) Whenever possible, upgrading only one intermediate system (bridge or router) at a time, confirming the success of the upgrade before upgrading additional systems.

In providing continued secure data connectivity while an in-service upgrade is performed, MKA makes a modest contribution to the task of upgrading network systems. MKA lacks the knowledge and scope to ensure or to check that best practices are being followed when the upgrade is being performed, and the network administrator is not relieved of these responsibilities.

The maximum time allowed for suspension, 120 seconds, is believed to be adequate for an upgrade, followed by expiry of the dead man timer, and reversion to the original software revision and configuration. However, if MKA successfully resumes operation, it could begin to distribute a new SAK at a time when the network operation center might not (for reasons completely unrelated to the use of MKA or MACsec) have re-established management connectivity with the upgrading system. If the system is then unilaterally suspended by operation of the dead man, it is possible that the members of the CAK will not all converge on use of the latest, or the old, SAK. The Key Server shall not redistribute the previous SAK (9.8). As a consequence, if a dead man timer or similar mechanism is being used by a resuming system, that system should not reset its suspendFor parameter to zero (see 9.18.4, 9.8) until the dead man timer has been reset.

MKA's in-service upgrade support can be deployed in environments where a comprehensive approach to system upgrade is already in place, and already synchronizes update and suspension activities. Such an existing approach can suspend MKA operation, as required, by coordinating the setting of suspendFor and suspendOnRequest parameters using each system's LMI. The values of suspendedWhile parameters for both Key Servers and other participants may also be set directly using the LMI, thus avoiding adding additional protocol dependencies to the existing coordination mechanism. Any such directly set values shall be consistent with the values that MKA would, and if not suspended will, communicate.

If the suspendFor or suspendedWhile timer values are set (either by using the MIB specified in Clause 13, or through the operation of other protocols) when there is no need for a suspension (i.e., the conditions for terminating the suspension are already satisfied) the suspension is terminated immediately and the values reset prior to including the timer value in any subsequent MKPDU. The conditions for such an immediate termination could occur as a result of one CA member initiating a suspension after other members have downloaded new software that enables them to upgrade during the suspension but before their own timer values are set.

9.18.7 MKPDU application data

Each participant that is capable of supporting in-service upgrades shall include the following parameter in each MKPDU transmitted (see Figure 11-16):

a) MKA suspension time.
The value transmitted is that of the suspendedWhile timer if the sending participant considers itself to be the Key Server for the MKA instance (i.e., has set bit 8 in octet 3 of the Basic Parameter Set, see Figure 11-8), and is the value of the suspendFor parameter otherwise.

and, when the Current Cipher Suite uses extended packet numbering, the following parameters:

b) The 32 most significant bits of the Lowest Acceptable PN for the Latest Key.
c) The 32 most significant bits of the Lowest Acceptable PN for the Old Key (if in use).

A receiving participant sets its member's suspendedWhile timer to a received suspendedWhile value iff it is the member's principal actor and agrees that the transmitter is the Key Server for its MKA instance.

A receiving participant records a suspendFor value received from any live partner, superseding any prior suspendFor value received from that partner.

9.19 In-service upgrade examples

This subclause (9.19) provides some examples of MKA operation, focusing on suspension. The parts of the MKA transport component of each MKPDU—the actor's member identifier and message number, and the member identifier and message number of each participant in the Live Peer List or Potential Peer List—and (some of) the other parameters transmitted are shown, as in 9.17, as follows:

Actor : Live Peer List : Potential Peer List: other parameters

where each MI, MN tuple is shown as X+1, X+2, etc. Initial MN values in these examples are arbitrary. Tuples in the peer lists, and other parameters, are separated by semi-colons.

9.19.1 Requested by end station in point-to-point CA

An end station port S, that is not the Key Server, requests suspension from the Key Server, K, by sending an MKPDU with a suspendFor value (encoded in the MKA Suspension Time field) of 60 seconds. K has suspendOnRequest True, and responds with an MKPDU with a suspendedWhile value (again encoded in the MKA Suspension Time field) of 60 seconds. On receipt, S suspends itself.

S_A	→ A+47:F+63::suspendFor = 60	→	K_F.. (1.1)
K_F	→ F+64:A+47::suspendedWhile = 60	→	S_A.. (1.2)
S_A	suspends		.. (1.3)

The next two or three periodic transmissions by K occur while S (as A) is still on K_F's Live Peer List, so the value of suspendedWhile remains at 60 seconds. The value of suspendedWhile in subsequent transmissions is decremented over time, but before it reaches zero (having reached 30 seconds in this example) S resumes with a new MI R. S has not kept accurate track of the duration of the suspension, and simply assumes a suspendedWhile value of 120 seconds, the maximum that can be requested.

K_F	→ F+65:A+47::suspendedWhile = 60	→	.. (1.4)
K_F	→ F+66:A+47::suspendedWhile = 60	→	.. (1.5)
K_F	→ F+67:::suspendedWhile = 58	→	.. (1.6)
...			
K_F	→ F+91:suspendedWhile = 30:	→	.. (1.7)
S_R	resumes, assuming suspendedWhile = 120		.. (1.8)

S then exchanges MKPDUs with K, and is recognized as a live peer. K terminates the suspension, and distributes a fresh SAK, with the key identifier (in this example) of F+2.

S_R	→ R+1:::suspendFor = 0	→	K_F.. (1.9)
K_F	→ F+92::R+1:suspendedWhile = 30	→	S_R..(1.10)
S_R	→ R+2:F+92::suspendFor = 0	→	K_F.. (1.9)

K_F	→ F+93:R+2::suspendedWhile = 0; DistribSAK = {SAK}F+2;
	SAKuse = F+1.0.rt, F+2.1.r
		→	S_R..(1.10)

S can then install and use the fresh key, exchanging MKPDUs as required (see 9.17 for relevant examples).

9.19.2 Initiated by Key Server in point-to-point CA

A Key Server initiates the suspension by sending an MKPDU with a suspendedWhile value (encoded, as always, in the MKA Suspension Time field) of 60 seconds, before suspending. In this particular example the other CA member takes the opportunity of suspending and upgrading itself at the same time.

K_F	→ F+64:A+47::suspendedWhile = 60	→	S_A.. (2.1)
S_A	suspends		.. (2.2)
K_F	→ F+65:A+47::suspendedWhile = 60	→	.. (2.3)
K_F	→ F+66:A+47::suspendedWhile = 60	→	.. (2.4)
K_F	suspends		.. (2.5)

K resumes, assuming an MI of D, and a suspendedWhile value of 120 seconds.

K_D	resumes, assuming suspendedWhile = 120		.. (2.6)
K_D	→ D+1:::suspendedWhile = 120	→	.. (2.7)

K_D → D+2:::suspendedWhile = 118 → .. (2.8)

S resumes, assuming an MI of G, and a suspendedWhile value of 120 seconds.

S_G resumes, assuming suspendedWhile = 120 .. (2.9)
S_G → G+1:::suspendFor = 0 → K_D..(2.10)
K_D → D+3::G+1:suspendedWhile = 118 → S_G..(2.11)
S_G → G+2:D+3::suspendFor = 0 → K_D..(2.12)
K_D → D+4:G+2::suspendedWhile = 0; DistribSAK = {SAK}D+1;
SAKuse = 0.0.rt, D+1.1.r → S_G..(2.13)

The installation and use of the fresh key now proceeds as before.

9.19.3 Intermediate systems suspending multiple CAs

An intermediate system, a router or bridge, will usually have to suspend MKA operation in the multiple CAs that it connects, as a control plane software upgrade will affect all of its ports. It might be the Key Server for some CAs and not for others. Careful planning is required (see 9.18.6) when upgrading intermediate systems in a network, and there are limits to the assistance and safeguards that can be provided by the operation of a local protocol, such as MKA. However, MKA, as shown in this deliberately complex example, does provide some help when multiple systems are involved and there has been a lack of coordination.

In this example, an intermediate system I has ports 1, 2, 3 with MKA participants I1, I2, I3, and neighbors A, B, C, respectively. I2 is already participating in a suspension initiated by B, which has suspended itself. The network administrator instructs I to upgrade, causing suspendFor to be set 60 seconds on each port. I1 is the Key Server for its CA, and can set and start transmitting suspendedWhile immediately. However it still has to arrange to suspend operation on its other ports, so it cannot suspend I1 immediately, so suspendFor and (as a consequence) suspendedWhile for I1 will remain at 60 seconds for the time being. I2 is also (in Bs absence) the Key Server for its CA, and can also set suspendedWhile directly. C, rather than I3, is the Key Server for the third port’s CA.

$I2_P$ suspendedWhile =15 .. (3.1)
I1 suspendFor = 60, I2 suspendFor = 60, 13 suspendFor = 60 .. (3.2)
$I1_J$ → J+31:V+60::suspendedWhile = 60 → A_V.. (3.3)
$I2_P$ → P+29:R+58::suspendedWhile = 60 → B_R.. (3.4)
$I3_Q$ → Q+33:W+62::suspendFor = 60 → C_W.. (3.5)

I might have to persist with these transmissions for some time, but if (and as soon as) C has suspendOnRequest set it will respond, and this can happen immediately.

C_W → W+63:Q+33::suspendedWhile = 60 → $I3_Q$.. (3.6)

I can then suspend and upgrade. While it is still suspended B might resume, but its newly assumed suspendedWhile value of 120 seconds will provide sufficient time for I2’s suspension (B was not operating when I2 suspended so has no information that would allow it to adopt a lower value).

B_K → K+1:::suspendedWhile = 120 → .. (3.7)

When I resumes, the participants for each of its ports and their partners will exchange MKPDUs as usual in order to recognize live peers, and fresh keys will be distributed.

9.19.4 Key Server suspends in a group CA

A, B, and C are members of a group CA. A, the Key Server, is to suspend to allow its control plane to be upgraded, and 30 seconds is believed to be sufficient for this to occur. The process is started by setting A's suspendFor parameter. This causes A to set its suspendedWhile parameter, and to transmit periodic MKPDUs for the next MKA Life Time (6 seconds) before suspending.

A_E → E+14:Y+15,L+14::suspendedWhile = 30 → B_Y, C_L.. (4.1)

For the following MKA Life Time A will remain on B and C's Live Peer Lists, so neither will claim to be the new Key Server. Finally, assuming that B has the higher Key Server Priority, B will become the Key Server and transmit the, by now decremented and continually decrementing, value of suspendedWhile in its periodic transmissions.

B_Y → Y+21:L+20::suspendedWhile = 24 → B_Y, C_L.. (4.1)

When A resumes, it will advertise suspendedWhile as 120 seconds, before exchanging MKPDUs and realizing that the conditions for terminating the suspension have been met. However, if A does not resume for any reason, B will terminate the suspension in 24 seconds.

11. EAPOL PDUs

11.5 EAPOL protocol version handling

Insert an additional note at the end of 11.5 as follows:

To ensure that backward compatibility is maintained between versions of this protocol, a version **A** protocol implementation shall interpret a received EAPOL PDU with protocol version number **B** as follows:

a) Where **B** is greater than or equal to **A**, the EAPOL PDU shall be interpreted as if it carried the supported version number, **A**, as follows:
   1) All parameters that are defined in version **A** shall be interpreted in the manner specified for version **A** of the protocol.
   2) All parameters not defined in version **A** for the given EAPOL Packet Type shall be ignored.
   3) All octets that appear in the EAPOL PDU beyond the largest numbered octet defined for version **A** for the received EAPOL Packet Type shall be ignored.

NOTE 1—As a consequence of these rules, a version 1 implementation ignores the version number. The rules allow future specification of protocol extensions, identified as new versions. Subsequent versions can be required to check the version number in order to correctly interpret the received PDU.

b) Where **B** is less than **A**, the EAPOL PDU shall be interpreted as specified for the version number, **B**, as follows:
   1) All parameters shall be interpreted in the manner specified for version **B** of the protocol.
   2) All parameters not defined in version **B** for the given EAPOL Packet Type shall be ignored.
   3) All octets that appear in the EAPOL PDU beyond the largest numbered octet defined for version **B** for the received EAPOL Packet Type shall be ignored.

NOTE 2—This edition of this standard provides all the information necessary to comply with the provisions of this subclause (11.5), without the need to consult prior editions for information on prior protocol versions.

NOTE 3—IEEE Std 802.1Xbx-2014 added optional support for in-service upgrades including suspension of MKA operation and recovery of the most significant bits of the PN for MACsec Cipher Suites that use Extended Packet Numbering. The EAPOL version number was unaffected by this amendment. Each MKPDU (an EAPOL PDU with a Packet Type of EAPOL-MKA) carries its own MKA Version Identifier (in the Basic Parameter Set, see 11.11, Figure 11-6, and Figure 11-8).

11.11 EAPOL-MKA

Change the existing note and insert an additional note in 11.11 as follows:

NOTE 1—This standard contains a number of provisions to guard against obsolescence by future developments in cryptography, without presuming to anticipate what those developments might be. These include the ability to select different ICV algorithms and sizes. The ICV will comprise the final octets of the Packet Body, whatever its size.

MKPDU encoding, validation, and decoding follows EAPOL's versioning rules (11.2, 11.5). The Basic Parameter Set includes an MKA Version Identifier that (with other parameters in the basic set) advertises the capabilities of the transmitting MKA implementation. This information can be supplemented both by version specific parameters within the basic set and by optional sets. A consistent TLV encoding identifies each set and allows it to be skipped if unrecognized by the receiver. Addition of parameters to existing sets, and the addition of parameter sets whose support is mandatory for a given version, will be accompanied by an MKA Version Identifier increment. This standard specifies the use of MKA Version Identifier ~~1~~ 2.

NOTE 2—IEEE Std 802.1Xbx-2014 added optional support for in-service upgrades including suspension of MKA operation and recovery of the most significant bits of the PN for MACsec Cipher Suites that use Extended Packet Numbering. The MKA Version Identifier was incremented to 2 by this amendment. A single optional parameter set was added, but there were also minor changes to the behavior of the CP state machine [as a consequence of changes to the specification of the state machine interface variable chgdServer (12.2)]. Those behavioral changes removed any need for a suspended system to record the identify of the Key Server specifically, and also avoid disrupting secure connectivity if another participant that is already a CA member takes over the role of Key Server; they are transparent to other CA members that are using MKA Version 1.

11.11.1 MKA parameter encoding

Change the third and fourth paragraphs of 11.11.1 and Table 11-7 as follows:

Table 11-7 specifies the parameter sets defined by this revision of this standard, the format and parameters for each set are specified in Figure 11-8 through Figure 11-13. Reserved bits within octets and reserved octets are shown as ‘X’ in the figures.

On receipt of an MKPDU, a PAE that transmits MKPDUs with a given MKA Version Identifier

a) Shall recognize and process each parameter set specified as mandatory for that version.

b) May recognize and process parameter sets specified as optional for that version.

c) Shall ignore any parameter set that is not specified as mandatory or optional for that version.

d) Shall recognize and process each of the parameters, within each parameter set processed, that are specified as mandatory for that version.

e) May recognize and process each of the parameters, within each parameter set processed, that are specified as optional for that version.

f) Shall ignore any parameter that is not specified as mandatory or optional for that version.

NOTE—The entries in Table 11-7 follow the EAPOL protocol version handling rules (11.5).

Table 11-7—MKPDU parameter sets

Parameter set and Parameter set type		Version		Parameters	Version		Parameter specification
		~~0~~1[a]	2		~~0~~1[a]	2	
Basic Parameter Set See Figure 11-8	—[b]	M	M	MKA Version Identifier	M	M	11.11
				Key Server Priority	M	M	9.5
				Key Server	M	M	9.5.1
				MACsec Desired	M	M	9.6.1
				MACsec Capability	M	M	9.6.1
				SCI	M	M	IEEE Std 802.1AE
				Actor’s Member Identifier	M	M	
				Actor’s Message Number	M	M	
				Algorithm Agility	M	M	
				CAK Name	M	M	9.3.1, 6.2.2, 6.3.3
Live Peer List See Figure 11-9	1	M	M	Member Identifier, Message Number tuples	M	M	~~9.9~~9.4.3
Potential Peer List See Figure 11-9	2	M	M	Member Identifier, Message Number tuples	M	M	~~9.9~~9.4.3

Table 11-7—MKPDU parameter sets (continued)

Parameter set and Parameter set type		Version		Parameters	Version		Parameter specification
		~~0~~1[a]	2		~~0~~1[a]	2	
MACsec SAK Use See Figure 11-10	3	M	M	Latest Key AN	M	M	9.8, 9.10
				Latest Key tx	M	M	9.10
				Latest Key rx	M	M	9.10
				Old Key AN	M	M	9.10
				Old Key tx	M	M	9.10
				Old Key rx	M	M	9.10
				Plain tx	M	M	—
				Plain rx	M	M	—
				Delay protect	M	M	9.10.1
				Latest Key Identifier (Key Server Member Identifier, Key Number)	M	M	9.8, 9.10.1
				Latest Key Lowest Acceptable PN	M	M	9.8, 9.10.1
				Old Key Identifier (Key Server Member Identifier, Key Number)	M	M	9.8, 9.10.1
				Old Key Lowest Acceptable PN	M	M	9.8, 9.10.1
Distributed SAK See Figure 11-11, Figure 11-12	4	M	M	AES Key Wrap of SAK	M	M	9.8
				Distributed AN	M	M	9.9
				Offset Confidentiality		M	9.7
				Key Number	M	M	9.8
				MACsec Cipher Suite	M	M	9.7
Distributed CAK See Figure 11-13	5	M	M	AES Key Wrap of CAK	M	M	9.5
				CA Key Name	M	M	9.3.1
KMD See Figure 11-14	6	M	M	KMD	M	M	12.6
Announcement ~~See Figure 11-14~~ See Figure 11-15	7	O[c]	O	Announcement TLVs	M	M	11.12
XPN	8	—	O[d]	MKA suspension time	—	M	9.18
				Latest Key: Lowest Acceptable PN (msbs)	—	M	
				Old Key: Lowest Acceptable PN (msbs)	—	M	
ICV Indicator ~~See Figure 11-16~~ See Figure 11-17		~~O~~ M[e]	M[e]	—	—		11.11.3,11.11.4

[a] M = mandatory to implement. O = optional. – = ignore on receipt.

[b] The Basic Parameter Set is identified by its position at the start of the MKPDU, the first octet encodes the MKA Version Identifier.

[c] Mandatory to implement if EAPOL-Announcements are sent [5.10 item i)].

[d] Mandatory to implement if support for Extended Packet Numbering is claimed (5.11.4).

[e] The ICV will not be encoded unless the Algorithm Agility parameter specifies the use of an ICV that is not 16 octets in length (11.11.3) and there is no requirement to implement such an algorithm; however, 11.11.4 states the requirement for processing the parameter set should it be received.

Change Figure 11-10 and the accompanying footnotes as follows:

Bit: 8	7	6	5	4	3	2	1	Octet:
Parameter set type = 3								1
Latest Key AN[a]		Latest Key tx	Latest Key rx	Old Key AN		Old Key tx	Old Key rx	2
Plain tx[b]	Plain rx[c]	X	Delay protect	Parameter set body length				3
Parameter set body length (cont)								4[d]
Latest Key: Key Server Member Identifier								5 – 16[d]
Latest Key: Key Number								17 – 20[d]
Latest Key: Lowest Acceptable PN[e]								21 – 24[d]
Old Key: Key Server Member Identifier								25 – 36[d]
Old Key: Key Number								37 – 40[d]
Old Key: Lowest Acceptable PN[e]								41 – 44[d]

Figure 11-10—MACsec SAK Use parameter set

[a] MKA uses the same AN for all the SAs for a given SAK, though IEEE Std 802.1AE does not impose this as a constraint.
[b] True if the associated Controlled Port is currently transmitting plain text, i.e., protectFrames (IEEE Std 802.1AE) is False.
[c] True if the associated Controlled Port is currently receiving plain text, i.e., validateFrames (IEEE Std 802.1AE) is not Strict.
[d] The parameter set body length will be 0 if MACsec is not supported and 40 otherwise.
[e] Least significant 32 bits if the MACsec Cipher Suite uses Extended Packet Numbering.

Change Figure 11-12 and the accompanying footnotes as follows:

Bit: 8	7	6	5	4	3	2	1	Octet:
Parameter set type = 4								1
Distributed AN[a]		Confidentiality Offset[b]		X	X	X	X	2
X	X	X	X	Parameter set body length				3
Parameter set body length (cont)								4[c]
Key Number								5 – 8
MACsec Cipher Suite								9 – 16[d]
AES Key Wrap of SAK as specified in 9.8								17 – 40[e]

Figure 11-12—Distributed SAK parameter set (other MACsec Cipher Suites)

[a] Set to zero if the Key Server has decided that MACsec is not to be used. Note 0 is a valid AN.
[b] Transmitted as zero and ignored on receipt if the Cipher Suite does not support Confidentiality Offset.
[c] The parameter set body length will be 0 if the Key Server has decided that plain text transmission, rather than MACsec should be used, to 28 if GCM-AES-128 (the default MACsec Cipher Suite) is being used (see Figure 11-11), and 36 or greater if the Cipher Suite reference number (IEEE Std 802.1AE-2006, 14.4) is explicitly included.
[d] Present iff the MACsec Cipher Suite is not GCM-AES-128 (Cipher Suite reference number 00-80-02-00-01-00-00-01).
[e] ~~This parameter set permits future specification of SAK distribution using other key wrap or secure formats of 24 octets or greater, including distribution of SAKs comprising more bits.Specification of additional parameter sets will be required if more than one key wrap is to be used with a given MACsec Cipher Suite.~~ The length shown denotes a wrapped 128-bit key.

Change Figure 11-13 and the accompanying footnotes as follows:

Bit:	8	7	6	5	4	3	2	1	Octet:
	Parameter set type = 5								1
	X	X	X	X	X	X	X	X	2
	X	X	X	X	Parameter set body length				3
	Parameter set body length (cont)								4
	AES Key Wrap of CAK as specified in 9.8								5 – 28[a],[b]
	CAK Key Name								29[b] –

Figure 11-13—Distributed CAK parameter set

[a]If a future specification requires distribution of a CAK using a different key wrap or secure format~~, or the distribution of a CAK comprising more bits,~~ an additional parameter set will be required.

[b]The length shown denotes a wrapped 128-bit key.

Insert a new Figure 11-16, renumbering subsequent figures as required, as follows:

Bit:	8	7	6	5	4	3	2	1	Octet:
	Parameter set type = 8								1
	MKA suspension time (seconds)								2[a]
	X	X	X	X	Parameter set body length				3
	Parameter set body length (cont)								4
	Latest Key: Lowest Acceptable PN (most significant 32 bits)[b]								5 – 8
	Old Key: Lowest Acceptable PN (most significant 32 bits)[b]								9 – 12

Figure 11-16—XPN parameter set

[a]The suspendedWhile timer value if the MKPDU has been transmitted by the Key Server, and the suspendFor parameter otherwise.

[b]Transmitted as zero, and ignored on receipt, if the MACsec Cipher Suite does not use Extended Packet Numbering.

Change subclause 11.12.3 as follows:

11.12.3 MACsec Cipher Suites TLV

The MACsec Cipher Suites TLV (Figure 11-21) contains a list of one or more Cipher Suites supported by the system (for access to the specified network if within a NID Set) transmitting the announcement. Each Cipher Suite in the list is represented by its 8 octet Cipher Suite reference number as specified by IEEE Std 802.1AE-2006 Clause 14. A 2 octet Cipher Suite dependent implementation capability field precedes each Cipher Suite reference number. If the Cipher Suite reference number identifies the Default Cipher Suite (GCM–AES–128, specified in IEEE Std 802.1AE-2006) or the GCM–AES-256 (specified in IEEE Std 802.1AEbn-2011), GCM–AES-XPN–128 or GCM–AES–XPN–256 Cipher Suite (specified in IEEE Std 802.1AEbw-2013), the two least significant bits of the implementation capability field encode the MACsec Capability parameter specified in Table 11-6 and the fourteen more significant bits are transmitted as 0 and ignored on receipt. If the Authentication Requirements TLV specifies support for MACsec, and the MACsec Cipher Suites TLV is not present for a given NID, or the TLV information string length is not a multiple of 10 octets, the recipient can assume that any Global MACsec Cipher Suites TLV applies to that NID. If no MACsec Cipher Suites TLV is encoded the recipient of the Announcement can assume that the Default Cipher Suite (specified in IEEE Std 802.1AE) is supported, with both integrity protection (without confidentiality) and integrity with confidentiality (with a confidentiality offset of 0), and is the only MACsec Cipher Suite supported. GCM–AES-XPN–128 and GCM–AES–XPN–256 do not support a confidentiality offset of other than 0.

12. PAE operation

Change the NOTE following the introductory text of Clause 12 as follows:

NOTE—In this clause all references of the form (1AE:n.n) are to clause n.n of IEEE Std 802.1AE-2006 as amended by IEEE Std 802.1AEbn-2011 and IEEE Std 802.1AEbw-2013.

12.1 Model of operation

Change the fourth paragraph of 12.1 as follows:

The Key Agreement Entity (KaY) manages the operation of zero or more MKA instances, each identified by a CKN and using a specified CAK, as specified in Clause 9. ~~Operation of each~~ The KaY's participation in a given instance is ~~initiated~~ represented by an actor created by the Logon Process (9.14, 12.2, 12.5)~~, through a call to the mka.authenticate procedure specifying a CAK and CKN (12.2)~~. An actor is considered successful if it has one or more live partners, and the actor or one of those partners has been elected Key Server for the MKA instance. The KaY can have multiple successful actors at any one time, but only one of these can be selected as the KaY's principal actor, responsible for controlling the associated PAC or SecY. The KaY determines the candidates for selection by comparing the priority of each of the Key Servers elected by successful actors. One of those Key Servers will have the highest priority. The candidates for selection as principal actor are limited to those successful actors that have elected that highest priority Key Server as their Key Server. If the KaY is that highest priority Key Server, it can choose any one of those candidates as its principal actor, and use it to distribute SAKs. If the KaY is not that highest priority Key Server, then its principal actor will be the candidate that has most recently received a Distributed SAK parameter set (see Figure 11-11, Figure 11-12) from that Key Server. Distributed SAK parameter sets received by actors that are not candidates for principal actor are ignored. The highest priority Key Server can change its principal actor (9.5), causing its peer KaYs to change their principal actors when a Distributed SAK parameter set is received.

MKA.secured (see 12.2) will be set iff the Key Server's principal actor has decided that MACsec is to be used and MKA.authenticated will be set iff it has decided on plain text transmission (Figure 11–12). ~~If the KaY succeeds in electing a Key Server and agreeing an SAK, it will report mka.authenticated for the CAK, CKN used by the highest priority Key Server.~~ The KaY ~~It~~ will also report ~~mka.failed~~ MKA.failed for any CKN for which it fails to find a partner or a Key Server (see 9.14). The Logon Process is also responsible for initiating EAP authentication as a Supplicant and or Authenticator if those capabilities are present and enabled, interfacing to the PACP state machines as specified in 8.4. CAKs that have been acquired from EAP may be cached in the CAK Cache. The CAK Cache may also be configured with one or more Pre-Shared Keys (PSKs). The Logon Process is thus responsible for the acquisition, use, and retention of all CAK, CKN tuples and for deciding (if necessary) that authentication or use of a mutual proof of prior authentication is not possible and selecting unauthenticated connectivity—provided that is permitted by CP controls. A pre-shared CAK, CKN and or CAK, CKN tuples from previously successful authentications can be used by the KaY at the same time as a fresh authentication attempt is made using EAP, or the latter can be delayed in anticipation that prior authentication result can be used. The choice of CAK, CKN tuples by the Logon Process may be guided by Network Identity information made available through EAPOL.

12.2 KaY interfaces

Change the first list item in 12.2 as follows:

— mka.enabled: Set by MKA if it is operational: enabled will be FALSE if the functionality provided by the PAE is not available, or not implemented, or the control variable mka.enable (see 9.16) has been cleared by management.

Change the sixth through ninth list items in 12.2, adding a further paragraph as follows:

- MKA.participate: Set by the Logon Process to ensure that the actor is an active participant ~~request the actor's active participation~~ in MKA. When set MKPDUs will be transmitted even if none are received and even if MKA.failed is set. When MKA.participate is not set, the actor will transmit only for a period of MKA Lifetime following the receipt of an MKPDU from a feasible partner (9.4.6). ~~Cleared by the Logon Process to request the actor to cease participation.~~
- MKA.authenticated: Set by MKA to indicate that the actor is the principal actor, i.e., is participating in the MKA instance that has elected the highest priority Key Server, and that Key Server has proved mutual authentication but has determined that Controlled Port communication should proceed without the use of MACsec (see 9.16).
- MKA.secured: Set by MKA to indicate that the actor is the principal actor, i.e., is participating in the MKA instance that has elected the highest priority Key Server, and that Key Server has specified the use of MACsec to secure communication (see 9.16).
- MKA.failed: Set by MKA to indicate that the actor has failed, i.e., the actor and its live partners (if any) do not include a participant willing to act as a Key Server. If the actor has failed but has not been deleted, it might receive further MKPDUs indicating ~~participate remains set, it will recommence participation if it receives an MKPDU (with the appropriate CKN) that indicates~~ that there is a potential partner and Key Server for the participants. In that case, MKA will clear failed.

  If in-service upgrades are supported (5.11.4, 9.18), and a suspension is in progress, the KaY will not reset MKA.secured (if set) until the suspension has been terminated (9.18.4). As a consequence (in the absence of further management changes, such as modification of the policy controls permitting connectivity) the Logon Process will not change the value of the connect signal to the CP state machine for the duration of the suspension and the MKA instance's Key Server (or its substitute) will not generate a fresh SAK, allowing the latter to remain in the CP:RETIRE or CP:SECURED states and provide continued data connectivity.

Change the tenth list item in 12.2 as follows:

- chgdServer: Set when a new Key Server, i.e., one whose SCI (9.4.4) was not among those used by existing SAs, has distributed an SAK ~~, has been elected, i.e., upon the first election and whenever there is a change in the Key Server as identified by its SCI (9.4.4)~~. Cleared by CP ~~when it takes note of the election~~.

12.5 Logon Process

Insert the following subclause title before the paragraph beginning "When and how connectivity will be provided ..." renumbering the existing subclause 12.5.1:

12.5.1 Controlling connectivity

Insert the following new subclause 12.5.2 and subclause title 12.5.3 before the paragraph beginning "The Logon Process may use Network Identities ..."

12.5.2 Active and passive participation

If an MKA instance is created (12.2) with a CAK derived from an EAP exchange and not previously cached, participate is set. If the KaY sets MKA.failed before the CAK is used successfully (setting MKA.authenticated or MKA.secured) the instance is deleted and the CAK discarded [see 9.14 (i)]. If the CAK is used successfully, then it will be cached (by default) with an activate (9.16) value of Default.

If an MKA instance is created with a CAK cached with an activate value of OnOperUp, participate is set for a period of MKA Life Time and then cleared. The participant will, therefore, remain active, transmitting MKPDUs, for at least MKA Life Time, but will become passive if it does not receive an MKPDU from a feasible partner for a period of MKA Life Time.

If an MKA instance is created with a CAK cached with an activate value of Always, participate is set and is not subsequently cleared even if no MKPDUs are received. The participant will remain persistently active, transmitting MKPDUs even in the prolonged absence of a feasible partner.

If an MKA instance was created from a cached CAK that has its activate value changed from OnOperUp to Always, then participate is set. If the value is changed from Always to OnOperUp, then participate is cleared.

12.5.3 Network Identities

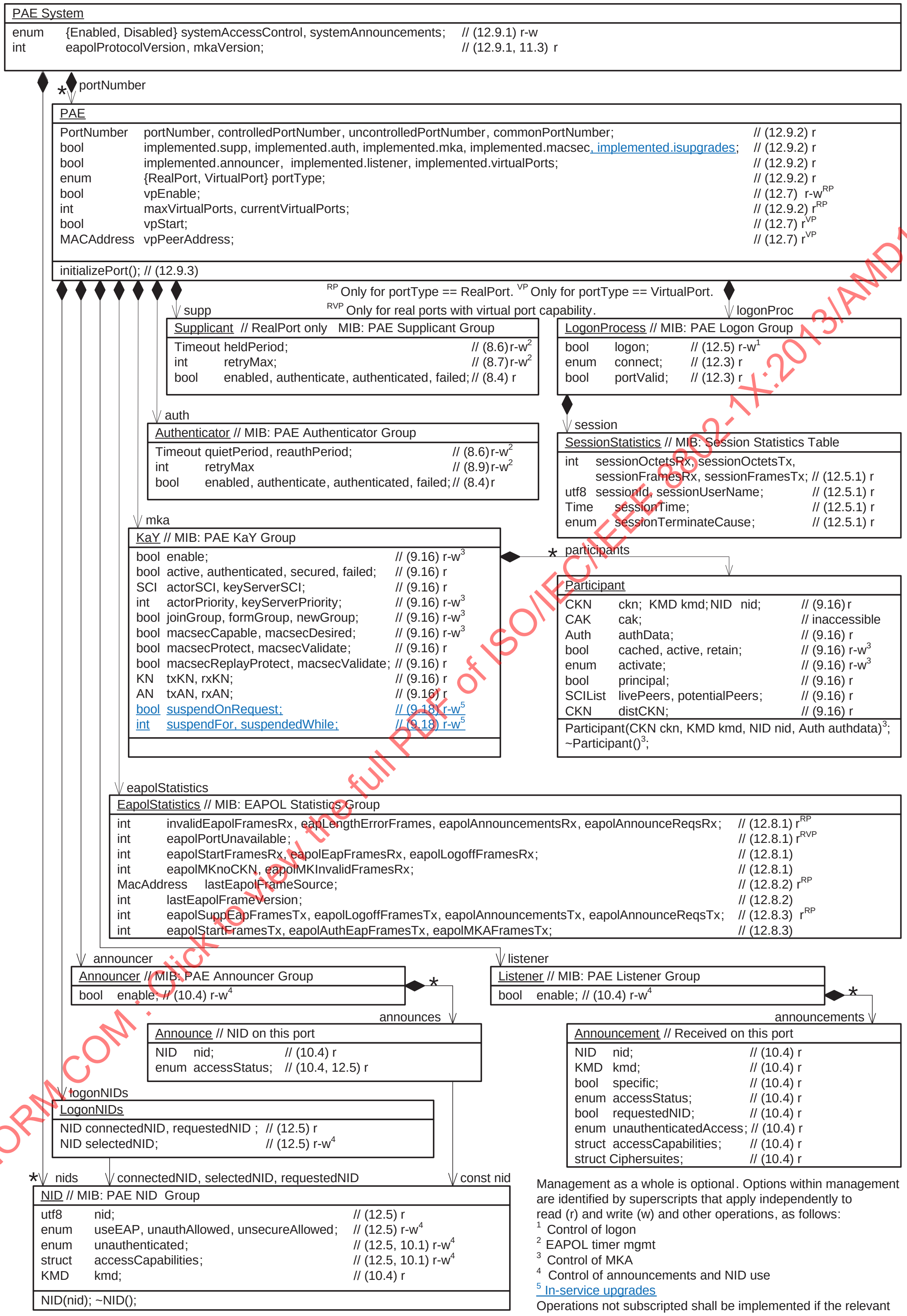
12.9 PAE management

12.9.2 Identifying PAEs and their capabilities

Add the following item to the dashed list following implemented.macsec:

— implemented.isupgrades: Set iff the MKA supports in-service upgrades (9.18).

Change Figure 12-3 to insert suspendOnRequest, suspendFor, suspendedWhile, and implemented.isupgrades as follows:


PAE System
enum {Enabled, Disabled} systemAccessControl, systemAnnouncements; // (12.9.1) r-w
int eapolProtocolVersion, mkaVersion; // (12.9.1, 11.3) r
*
portNumber
PAE
PortNumber portNumber, controlledPortNumber, uncontrolledPortNumber, commonPortNumber; // (12.9.2) r
bool implemented.supp, implemented.auth, implemented.mka, implemented.macsec, implemented.isupgrades; // (12.9.2) r
bool implemented.announcer, implemented.listener, implemented.virtualPorts; // (12.9.2) r
enum {RealPort, VirtualPort} portType; // (12.9.2) r
bool vpEnable; // (12.7) r-w RP
int maxVirtualPorts, currentVirtualPorts; // (12.9.2) r RP
bool vpStart; // (12.7) r VP
MACAddress vpPeerAddress; // (12.7) r VP
initializePort(); // (12.9.3)
RP Only for portType == RealPort. VP Only for portType == VirtualPort.
RVP Only for real ports with virtual port capability.
supp
Supplicant // RealPort only MIB: PAE Supplicant Group
Timeout heldPeriod; // (8.6) r-w 2
int retryMax; // (8.7) r-w 2
bool enabled, authenticate, authenticated, failed; // (8.4) r
logonProc
LogonProcess // MIB: PAE Logon Group
bool logon; // (12.5) r-w 1
enum connect; // (12.3) r
bool portValid; // (12.3) r
auth
Authenticator // MIB: PAE Authenticator Group
Timeout quietPeriod, reauthPeriod; // (8.6) r-w 2
int retryMax // (8.9) r-w 2
bool enabled, authenticate, authenticated, failed; // (8.4) r
session
SessionStatistics // MIB: Session Statistics Table
int sessionOctetsRx, sessionOctetsTx,
sessionFramesRx, sessionFramesTx; // (12.5.1) r
utf8 sessionId, sessionUserName; // (12.5.1) r
Time sessionTime; // (12.5.1) r
enum sessionTerminateCause; // (12.5.1) r
mka
KaY // MIB: PAE KaY Group
bool enable; // (9.16) r-w 3
bool active, authenticated, secured, failed; // (9.16) r
SCI actorSCI, keyServerSCI; // (9.16) r
int actorPriority, keyServerPriority; // (9.16) r-w 3
bool joinGroup, formGroup, newGroup; // (9.16) r-w 3
bool macsecCapable, macsecDesired; // (9.16) r-w 3
bool macsecProtect, macsecValidate; // (9.16) r
bool macsecReplayProtect, macsecValidate; // (9.16) r
KN txKN, rxKN; // (9.16) r
AN txAN, rxAN; // (9.16) r
bool suspendOnRequest; // (9.18) r-w 5
int suspendFor, suspendedWhile; // (9.18) r-w 5
*
participants
Participant
CKN ckn; KMD kmd; NID nid; // (9.16) r
CAK cak; // inaccessible
Auth authData; // (9.16) r
bool cached, active, retain; // (9.16) r-w 3
enum activate; // (9.16) r-w 3
bool principal; // (9.16) r
SCIList livePeers, potentialPeers; // (9.16) r
CKN distCKN; // (9.16) r
Participant(CKN ckn, KMD kmd, NID nid, Auth authdata) 3;
~Participant() 3;
eapolStatistics
EapolStatistics // MIB: EAPOL Statistics Group
int invalidEapolFramesRx, eapLengthErrorFrames, eapolAnnouncementsRx, eapolAnnounceReqsRx; // (12.8.1) r RP
int eapolPortUnavailable; // (12.8.1) r RVP
int eapolStartFramesRx, eapolEapFramesRx, eapolLogoffFramesRx; // (12.8.1)
int eapolMKnoCKN, eapolMKInvalidFramesRx; // (12.8.1)
MacAddress lastEapolFrameSource; // (12.8.2) r RP
int lastEapolFrameVersion; // (12.8.2)
int eapolSuppEapFramesTx, eapolLogoffFramesTx, eapolAnnouncementsTx, eapolAnnounceReqsTx; // (12.8.3) r RP
int eapolStartFramesTx, eapolAuthEapFramesTx, eapolMKAFramesTx; // (12.8.3)
announcer
Announcer // MIB: PAE Announcer Group
bool enable; // (10.4) r-w 4
*
announces
Announce // NID on this port
NID nid; // (10.4) r
enum accessStatus; // (10.4, 12.5) r
listener
Listener // MIB: PAE Listener Group
bool enable; // (10.4) r-w 4
*
announcements
Announcement // Received on this port
NID nid; // (10.4) r
KMD kmd; // (10.4) r
bool specific; // (10.4) r
enum accessStatus; // (10.4) r
bool requestedNID; // (10.4) r
enum unauthenticatedAccess; // (10.4) r
struct accessCapabilities; // (10.4) r
struct Ciphersuites; // (10.4) r
logonNIDs
LogonNIDs
NID connectedNID, requestedNID ; // (12.5) r
NID selectedNID; // (12.5) r-w 4
*
nids
connectedNID, selectedNID, requestedNID
const nid
NID // MIB: PAE NID Group
utf8 nid; // (12.5) r
enum useEAP, unauthAllowed, unsecureAllowed; // (12.5) r-w 4
enum unauthenticated; // (12.5, 10.1) r-w 4
struct accessCapabilities; // (12.5, 10.1) r-w 4
KMD kmd; // (10.4) r
NID(nid); ~NID();

Management as a whole is optional. Options within management are identified by superscripts that apply independently to read (r) and write (w) and other operations, as follows:

[1] Control of logon
[2] EAPOL timer mgmt
[3] Control of MKA
[4] Control of announcements and NID use
[5] In-service upgrades

Operations not subscripted shall be implemented if the relevant capability is identified as implemented by the PAE.

Figure 12-3—PAE management information

13. PAE MIB

Change the following row in Table 13-4 after rxAN:

Table 13-4—PAE managed object cross-reference table

PAE management information (Figure 12-3)	MIB object(s)
macsecCapable (9.16) r-w	ieee8021XKayMacSecCapab~~le~~ility

Insert the following rows into Table 13-4 after rxAN:

Table 13-4—PAE managed object cross-reference table

PAE management information (Figure 12-3)	MIB object(s)
suspendFor (9.16) r-w	ieee8021XKayMkaSuspendFor
suspendOnRequest (9.16) r-w	ieee8021XKayMkaSuspendOnRequest
suspendedWhile (9.16) r-w	ieee8021XKayMkaSuspendedWhile

13.4 Security considerations

Insert the following NOTE after the last paragraph of 13.4:

NOTE—The IEEE 802.1Xbx amendment to this standard added the in-service upgrade (9.18) group (ieee8021XPaeKaYIsupgradeGroup). This addition does not affect the security considerations to be taken into account when making use of this standard.

13.5 Definitions for PAE MIB

Delete the entire text of the MIB definition, following the introductory sentence, and insert the following text:

```
-- **********************************************************************
--
-- IEEE8021X-PAE-MIB : MIB for IEEE 802.1X (802.1X-2010 + 802.1Xbx)
--
-- **********************************************************************

IEEE8021X-PAE-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Gauge32,
    Counter32,
    Counter64,
    Unsigned32,
    Integer32
        FROM SNMPv2-SMI
    MacAddress,
    TEXTUAL-CONVENTION,
    TruthValue,
    RowPointer,
    TimeStamp,
    TimeInterval,
    RowStatus
        FROM SNMPv2-TC
    MODULE-COMPLIANCE,
    OBJECT-GROUP
```

```
        FROM SNMPv2-CONF
    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB
    InterfaceIndex
        FROM IF-MIB
    SecySCI
        FROM IEEE8021-SECY-MIB;

ieee8021XPaeMIB MODULE-IDENTITY
    LAST-UPDATED    "201404101619Z"
    ORGANIZATION    "IEEE 802.1 Working Group"
    CONTACT-INFO
      " WG-URL: http://grouper.ieee.org/groups/802/1/index.html
        WG-EMail: stds-802-1@ieee.org
        Contact: Mick Seaman
        Postal: C/O IEEE 802.1 Working Group
                IEEE Standards Association
                445 Hoes Lane
                P.O. Box 1331
                Piscataway
                NJ 08855-1331
                USA
                E-mail: STDS-802-1-L@LISTSERV.IEEE.ORG"
    DESCRIPTION
        "The MIB module for managing the Port Access Entity (PAE)
        functions of IEEE 802.1X (Revision of 802.1X-2004).
        The PAE functions managed are summarized in Figure 12-3 of
        IEEE 802.1X and include EAPOL PACP support for authentication
        (EAP Supplicant and/or Authenticator), MACsec Key Agreement
        (MKA), EAPOL, and transmission and reception of network
        announcements.

        The following acronyms and definitions are used in this MIB.

        AN : Association Number, a number that is concatenated with a
            MACsec Secure Channel Identifier to identify a Secure
            Association (SA).

        Announcer : EAPOL-Announcement transmission functionality.

        Authenticator : An entity that facilitates authentication of
            other entities attached to the same LAN.

        CA : secure Connectivity Association: A security relationship,
            established and maintained by key agreement protocols, that
            comprises a fully connected subset of the service access
            points in stations attached to a single LAN that are to be
            supported by MACsec.

        CAK : secure Connectivity Association Key, a secret key
            possessed by members of a given CA.

        CKN : secure Connectivity Association Key Name (CKN), a text
            that identifies a CAK.

        Common Port : An instance of the MAC Internal Sublayer Service
            used by the SecY or PAC to provide transmission and
            reception of frames for both the Controlled and
            Uncontrolled Ports.

        Controlled Port : The access point used to provide the secure
            MAC Service to a client of a PAC or SecY.

        CP state machine : Controlled Port state machine is capable of
            controlling a SecY or a PAC.  The CP supports
            interoperability with unauthenticated systems that are not
            port-based network access control capable, or that lack
            MKA.  When the access controlled port is supported by a
            SecY, the CP is capable of controlling the SecY so as to
            provide unsecured connectivity to systems that implement a
            PAC.
```

```
EAP : Extensible Authentication Protocol, RFC3748.

EAPOL : EAP over LANs.

KaY : Key Agreement Entity, a PAE entity responsible for MKA.

Key Server : Elected by MKA, to transport a succession of SAKs,
    for use by MACsec, to the other member(s) of a CA.

KMD : Key Management Domain, a string identifying systems that
     share cached CAKs.

Listener : The role is to receive the network announcement
    parameters in the authentication process.

Logon Process : The Logon Process is responsible for the
    managing the use of authentication credentials, for
    initiating use of the PAE’s Supplicant and or Authenticator
    functionality, for deriving CAK, CKN tuples from PAE
    results, for maintaining PSKs (Pre-Sharing Keys), and for
    managing MKA instances.  In the absence of successful
    authentication, key agreement, or support for MAC Security,
    the Logon Process determines whether the CP state machine
    should provide unauthenticated connectivity or
    authenticated but unsecured connectivity.

MKA : MACsec Key Agreement protocol allows PAEs, each
    associated with a port that is an authenticated member of a
    secure connectivity association (CA) or a potential CA, to
    discover other PAEs attached to the same LAN, to confirm
    mutual possession of a CAK and hence to prove a past mutual
    authentication, to agree the secret keys (SAKs) used by
    MACsec for symmetric shared key cryptography, and to ensure
    that the data protected by MACsec has not been delayed.

MKPDU : MACsec Key Agreement Protocol Data Unit.

MPDU : MAC Protocol Data Unit.

NID : Network Identity, a UTF-8 string identifying an network
     or network service.

PAE : Port Access Entity, the protocol entity associated with a
     Port.  It can support the protocol functionality
     associated with the Authenticator, the Supplicant, or
     both.

PAC : Port Access Controller, a protocol-less shim that
     provides control over frame transmission and reception by
     clients attached to its Controlled Port, and uses the MAC
     Service provided by a Common Port.  The access control
     decision is made by the PAE, typically taking into
     account the success or failure of mutual authentication
     and authorization of the PAE’s peer(s), and is
     communicated by the PAE using the LMI to set the PAC’s
     Controlled Port enabled/disable.  Two different interfaces
     ‘Controlled Port’ and ‘Uncontrolled Port’, are associated
     with a PAC, and that for each instance of a PAC, two
     ifTable rows (one for each interface) run on top of an
     ifTable row representing the ‘Common Port’ interface,
     such as a row with ifType = ‘ethernetCsmacd(6)’.

     For example :
     ----------------------------------------------------------
    |                            |                            |
    |   Controlled Port          |   Uncontrolled Port        |
    |      Interface             |      Interface             |
    |    (ifEntry = j)           |     (ifEntry = k)          |
    | (ifType =                  | (ifType =                  |
    |  macSecControlledIF(231))  |  macSecUncontrolledIF(232))|
    |                            |                            |
    |---------------------------------------------------------|
        |                                                          |
        |                    Physical Interface                    |
        |                      (ifEntry = i)                       |
        |                (ifType = ethernetCsmacd(6))              |
        |__________________________________________________________|
                    i, j, k are ifIndex to indicate
                   an interface stack in the ifTable.
                    Figure : PAC Interface Stack

         The 'Controlled Port' is the service point to provide one
         instance of the secure MAC service in a PAC.  The
         'Uncontrolled Port' is the service point to provide one
         instance of the insecure MAC service in a PAC.

    PACP : Port Access Controller Protocol.

    Port Identifier : A 16-bit number that is unique within the
        scope of the address of the port.

    Real Port : Indicates the PAE is for a real port.  A port that
        is not created on demand by the mechanisms specified in
        this standard, but that can transmit and receive frames for
        one or more virtual ports.

    SC : Secure Channel, a security relationship used to provide
        security guarantees for frames transmitted from one member
        of a CA to the others.  An SC is supported by a sequence of
        SAs thus allowing the periodic use of fresh keys without
        terminating the relationship.

    SA : Secure Association, a security relationship that provides
        security guarantees for frames transmitted from one member
        of a CA to the others. Each SA is supported by a single
        secret key, or a single set of keys where the cryptographic
        operations used to protect one frame require more than one
        key.

    SAK : Secure Association key, the secret key used by an SA.

    SCI : Secure Channel Identifier, a globally unique identifier
        for a secure channel, comprising a globally unique MAC
        Address and a Port Identifier, unique within the system
        allocated that address.

    secured connectivity : Data transfer between two or 'Controlled
        Ports' that is protected by MACsec.

    SecY : MAC Security Entity, the entity that operates the MAC
        Security protocol within a system.

    Supplicant : An entity at one end of a point-to-point LAN
        segment that seeks to be authenticated by an Authenticator
        attached to the other end of that link.

    Suspension: Temporary suspension of MKA operation to facilitate
        in-service control plane software upgrades without
        disrupting existing secure connectivity.

    Uncontrolled Port : The access point used to provide the
        insecure MAC Service to a client of a SecY or PAC.

    Virtual Port : Indicates the PAE is for a virtual port.  A MAC
        Service or Internal Sublayer service access point that is
        created on demand.  Virtual ports can be used to provide
        separate secure connectivity associations over the same
        LAN."
REVISION        "201404101619Z"
DESCRIPTION
    "Update published as part of IEEE 802.1Xbx (Amendment to
    IEEE 802.1X-2010)"
REVISION        "200910011650Z"
DESCRIPTION
```

```
        "Initial version of this MIB module.  Published as part of
        IEEE P802.1X (Revision of IEEE Standard 802.1X-2009)"
    ::= { iso(1) iso-identified-organization(3) ieee(111)
          standards-association-numbered-series-standards(2)
          lan-man-stds(802) ieee802dot1(1) ieee802dot1mibs(1) 15 }

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

Ieee8021XPaeCKN ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This textual convention indicates the CAK name to identify
        the Connectivity Association Key (CAK) which is the root key
        in the MACsec Key Agreement key hierarchy.  All potential
        members of the CA use the same CKN."

    REFERENCE       "IEEE 802.1X Clause 5.4, Clause 9.3.1, Clause 6.2"
    SYNTAX          OCTET STRING (SIZE (1..16))

Ieee8021XPaeCKNOrNull ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This textual convention indicates the CAK name to identify
        the Connectivity Association Key (CAK) which is the root key
        in the MACsec Key Agreement key hierarchy.  All potential
        members of the CA use the same CKN.

        If this is a zero length value, then the NULL string means
        CKN information is applicable."

    REFERENCE       "IEEE 802.1X Clause 5.4, Clause 9.3.1, Clause 6.2"
    SYNTAX          OCTET STRING (SIZE (0..16))

Ieee8021XPaeKMD ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This textual convention indicates a Key Management Domain
        (KMD).

        KMD is a string of UTF-8 characters that names the transmitting
        authenticator's key management domain."

    REFERENCE       "IEEE 802.1X Clause 12.6"
    SYNTAX          OCTET STRING (SIZE (0..253))

Ieee8021XPaeNID ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This textual convention indicates a Network Identifier (NID).

        Each network is identified by a NID, a UTF-8 string used by
        network attached systems to select a network profile."

    REFERENCE       "IEEE 802.1X Clause 12.6, Clause 10.1"
    SYNTAX          OCTET STRING (SIZE (1..100))

Ieee8021XPaeNIDOrNull ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        "This textual convention indicates a Network Identifier (NID).

        Each network is identified by a NID, a UTF-8 string used by
        network attached systems to select a network profile.

        If this is a zero length value, then the NULL string for
        NID information is applicable."

    REFERENCE       "IEEE 802.1X Clause 12.6, Clause 10.1"
    SYNTAX          OCTET STRING (SIZE (0..100))
```

```
Ieee8021XMkaKeyServerPriority ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        “This textual convention indicates a Key Server priority
        information.

        Each MKA participant encodes a Key Server Priority, an 8-bit
        integer, in each MKPDU.  Each participant selects the live
        participant advertising the highest priority as its Key Server
        provided that participant has not selected another as its Key
        Server or is unwilling to act as the Key Server.  If a Key
        Server cannot be selected SAKs are not distributed.  In the
        event of a tie for highest priority Key Server, the member with
        the highest priority SCI is chosen.  For consistency with other
        uses of the SCI’s MAC Address component as a priority,
        numerically lower values of the Key Server Priority and SCI are
        accorded the highest priority.  The Table 9-2 contains
        recommendations for the use of priority values for various
        system roles. Participants that will never act as a Key Server
        should advertise priority 0xFF.”

    REFERENCE       “IEEE 802.1X Clause 9.5, Table 9-2”
    SYNTAX          OCTET STRING (SIZE (1))

Ieee8021XMkaMI ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        “This textual convention indicates a Member Identifier (MI).

        The MI is a 96-bit random value chosen when the MKA Instance
        begins, used with a 32-bit MN to protect against replay attacks
        and to record liveliness in the Live Peer List or potential
        liveliness in the Potential Peer List. If the MN wraps, a new
        random MI value is chosen and the MN begins again at 1.”

    REFERENCE       “IEEE 802.1X Clause 9.4.2”
    SYNTAX          OCTET STRING (SIZE (12))

Ieee8021XMkaMN ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    “d”
    STATUS          current
    DESCRIPTION
        “This textual convention indicates a Member Number (MN).

        The MN is a 32-bit value which begins at 1 and increases for
        each MKPDU transmitted.  It is used with the MI to protect
        against replay attacks and to record liveliness in the Live
        Peers List or potential liveliness in the Potential Peer List.
        If the MN wraps, a new random MI value is chosen and the MN
        begins again at a value of 1.”

    REFERENCE       “IEEE 802.1X Clause 9.4.2”
    SYNTAX          Unsigned32 (1..2147483648)

Ieee8021XMkaKN ::= TEXTUAL-CONVENTION
    DISPLAY-HINT    “d”
    STATUS          current
    DESCRIPTION
        “This textual convention indicates a Key Number (KN) used in
        MKA.

        The MN is a 32-bit integer assigned by that Key Server
        (sequentially, beginning with 1).”

    REFERENCE       “IEEE 802.1X Clause 9.8”
    SYNTAX          Unsigned32 (1..2147483648)

Ieee8021XPaeNIDCapabilites ::= TEXTUAL-CONVENTION
    STATUS          current
    DESCRIPTION
        “This textual convention indicates the combinations of
        authentication and protection capabilities supported for a
```

```
          NID. Any set of these combinations can be supported."

     REFERENCE        "IEEE 802.1X Clause 10.1, Table 11-8"
     SYNTAX           BITS {
                          eap(0),
                          eapMka(1),
                          eapMkaMacSec(2),
                          mka(3),
                          mkaMacSec(4),
                          higherLayer(5), -- WebAuth
                          higherLayerFallback(6), -- WebAuth
                          vendorSpecific(7)
                      }

Ieee8021XPaeNIDAccessStatus ::= TEXTUAL-CONVENTION
     STATUS           current
     DESCRIPTION
          "This textual convention indicates the transmitter's
          Controlled Port operational status and current level of
          access resulting from authentication and the consequent
          authorization controls applied by that port's clients.

          'noAccess' : Other than to authentication services, and to
              services announced as available in the absence of
              authentication (unauthenticated).

          'remedialAccess' : The access granted is severely limited,
              possibly to remedial services.

          'restrictedAccess' : The Controlled Port is operational, but
              restrictions have been applied by the network that can
              limit access to some resources.

          'expectedAccess' : The Controlled Port is operational, and
              access provided is as expected for successful
              authentication and authorization for the NID."

     REFERENCE        "IEEE 802.1X Clause 10.1, Table 11-8"
     SYNTAX           INTEGER  {
                          noAccess(0),
                          remedialAccess(1),
                          restrictedAccess(2),
                          expectedAccess(3)
                      }

Ieee8021XPaeNIDUnauthenticatedStatus ::= TEXTUAL-CONVENTION
     STATUS           current
     DESCRIPTION
          "This textual convention indicates the access capabilities of
          the port's clients without authentication.

          'noAccess' : Other than to authentication services (see
              Ieee8021XPaeNIDCapabilites information.

          'fallbackAccess' : Limited access can be provided after
              authentication failure.

          'limitedAccess' : Immediate limited access is available
              without authentication.

          'openAccess' : Immediate access is available without
              authentication."

     REFERENCE        "IEEE 802.1X Clause 10.1, Table 11-8"
     SYNTAX           INTEGER  {
                          noAccess(0),
                          fallbackAccess(1),
                          limitedAccess(2),
                          openAccess(3)
                      }

-- ------------------------------------------------------------------ --
-- Groups in the IEEE 802.1X MIB
-- ------------------------------------------------------------------ --

ieee8021XPaeMIBNotifications  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIB 0 }

ieee8021XPaeMIBObjects  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIB 1 }

ieee8021XPaeMIBConformance  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIB 2 }

-- ------------------------------------------------------------------ --
-- Management Objects in the IEEE 802.1X MIB
-- ------------------------------------------------------------------ --

ieee8021XPaeSystem  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 1 }

ieee8021XPaeLogon  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 2 }

ieee8021XPaeAuthenticator  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 3 }

ieee8021XPaeSupplicant  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 4 }

ieee8021XPaeEapol  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 5 }

ieee8021XPaeKaY  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 6 }

ieee8021XPaeNetworkIdentifier  OBJECT IDENTIFIER
    ::= { ieee8021XPaeMIBObjects 7 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE System Group
-- ------------------------------------------------------------------ --
--
-- ------------------------------------------------------------------ --
-- The 802.1X PAE System Objects
-- ------------------------------------------------------------------ --

ieee8021XPaeSysAccessControl OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object enables or disables port-based network access
        control for all the system’s ports.  Setting this control
        object to ‘false’ causes the following actions :
            . Deletes any virtual ports previously instantiated.
            . Terminates authentication exchanges and MKA instances’
              operation.
            . Each real port PAE behaves as if no virtual ports
              created.
            . All the PAEs’ Supplicant, Authenticator, and KaY are
              disabled.
            . Logon Process(es) behave as if the object
              ieee8021XNidUnauthAllowed was ‘immediate’.
            . Announcements can be transmitted, both periodically and
              in response to announcement requests (conveyed by
              EAPOL-Starts or EAPOL-Announcement-Reqs) but are sent
              with a single NULL NID.
            . Objects announcementAccessStatus and announceAccessStatus
              have the ‘noAccess’ value, announcementAccessRequested is
              ‘false’, object announcementUnauthAccess has the
              ‘openAccess’ value.
        The control variable settings for each real port PAE in the
        ieee8021XPaePortTable are unaffected, and will be used once the
        object is set to ‘true’.

        This configured value for this object shall be stored in
        persistent memory and remain unchanged across a
        re-initialization of the management system of the entity.”
    REFERENCE
        “IEEE 802.1X Clause 12.9.1, Figure 12-3 PAE
            System.systemAccessControl”
    ::= { ieee8021XPaeSystem 1 }

ieee8021XPaeSysAnnouncements OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Setting this control object to ‘false’ causes each PAE in this
        system to behave as if the PAE’s Announcement functionality is
        disabled.  The independent controls for each PAE apply if
        this object is ‘true’.

        This configured value for this object shall be stored in
        persistent memory and remain unchanged across a
        re-initialization of the management system of the entity.”
    REFERENCE
        “IEEE 802.1X Clause 12.9.1, Figure 12-3 PAE
             System.systemAnnouncements”
    ::= { ieee8021XPaeSystem 2 }

ieee8021XPaeSysEapolVersion OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The EAPOL protocol version for this system.”
    REFERENCE
        “IEEE 802.1X Clause 12.9.1, Clause 11.3, Figure 12-3 PAE
             System.eapolProtocolVersion”
    ::= { ieee8021XPaeSystem 3 }

ieee8021XPaeSysMkaVersion OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The MKA protocol version for this system.”
    REFERENCE       “IEEE 802.1X Clause 12.9.1”
    ::= { ieee8021XPaeSystem 4 }
-- ------------------------------------------------------------------ --
-- The 802.1X PAE Port Table
-- ------------------------------------------------------------------ --

ieee8021XPaePortTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XPaePortEntry
    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.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3 PAE”
    ::= { ieee8021XPaeSystem 5 }

ieee8021XPaePortEntry OBJECT-TYPE
    SYNTAX          Ieee8021XPaePortEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “The Port number, protocol version, and
        initialization control for a Port.

         If the PAE has been dynamically instantiated to support an
         existing or potential virtual port, the Uncontrolled Port
         interface and Controlled Port interface are allocated by the
         real port’s PAE.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XPaePortTable 1 }

Ieee8021XPaePortEntry ::= SEQUENCE {
        ieee8021XPaePortNumber              InterfaceIndex,
        ieee8021XPaePortType                INTEGER,
        ieee8021XPaeControlledPortNumber    InterfaceIndex,
        ieee8021XPaeUncontrolledPortNumber  InterfaceIndex,
        ieee8021XPaeCommonPortNumber        InterfaceIndex,
        ieee8021XPaePortInitialize          TruthValue,
        ieee8021XPaePortCapabilities        BITS,
        ieee8021XPaePortVirtualPortsEnable  TruthValue,
        ieee8021XPaePortMaxVirtualPorts     Unsigned32,
        ieee8021XPaePortCurrentVirtualPorts Gauge32,
        ieee8021XPaePortVirtualPortStart    TruthValue,
        ieee8021XPaePortVirtualPortPeerMAC  MacAddress,
        ieee8021XPaePortLogonEnable         TruthValue,
        ieee8021XPaePortAuthenticatorEnable TruthValue,
        ieee8021XPaePortSupplicantEnable    TruthValue,
        ieee8021XPaePortKayMkaEnable        TruthValue,
        ieee8021XPaePortAnnouncerEnable     TruthValue,
        ieee8021XPaePortListenerEnable      TruthValue
}

ieee8021XPaePortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An interface index indicates the port number associated with
        this port.  Each PAE is uniquely identified by a port number.
        The port number used is unique amongst all port numbers for
        the system, and directly or indirectly identifies the
        Uncontrolled Port that supports the PAE.

        If the PAE indicates a real port, ieee8021XPaePortType object
        in the same row is ‘realPort’, the port number shall be the
        same as the ieee8021XPaeCommonPortNumber object in the same row
        for the associated PAC or SecY.

        If the PAE indicates a virtual port, ieee8021XPaePortType
        object in the same row is ‘virtualPort’, this port number
        should be the same as the uncontrolledPortNumber object in the
        same row for the associated PAC or SecY.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 1 }

ieee8021XPaePortType OBJECT-TYPE
    SYNTAX          INTEGER  {
                        realPort(1),
                        virtualPort(2)
                    }
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The port type of the PAE.

        realPort(1) : indicates the PAE is for a real port.

        virtualPort(2) : indicates the PAE is for a virtual port.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 2 }
```

```
ieee8021XPaeControlledPortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “An interface index indicates the port number associated with
        PAC or SecY’s Controlled Port.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 3 }

ieee8021XPaeUncontrolledPortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “An interface index indicates the port number associated with
        PAC or SecY’s Uncontrolled Port.  If the PAE supports a
        real port, this port number can be the same as the
        ieee8021XPaeCommonPortNumber object in the same row, otherwise
        it shall not be the same.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 4 }

ieee8021XPaeCommonPortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “An interface index indicates the port number associated with
        PAC or SecY’s ‘Common Port’.  All the virtual ports created
        for a given real port share the same ‘Common Port’ and
        ieee8021XPaeCommonPortNumber in the same row.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 5 }

ieee8021XPaePortInitialize OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “The initialization control for this Port. Setting this object
        ‘true’ causes the Port to be reinitialized, terminating (and
        potentially restarting) authentication exchanges and MKA
        operation.

        If the port is a real port, any virtual ports previously
        instantiated are deleted.  Virtual ports can be reinstantiated
        through normal protocol operation.

        The object value reverts to ‘false’ once initialization
        has completed.”
    REFERENCE       “802.1X Clause 12.9.3, Figure 12-3”
    ::= { ieee8021XPaePortEntry 6 }

ieee8021XPaePortCapabilities OBJECT-TYPE
    SYNTAX          BITS {
                        suppImplemented(0),
                        authImplemented(1),
                        mkaImplemented(2),
                        macsecImplemented(3),
                        announcementsImplemented(4),
                        listenerImplemented(5),
                        virtualPortsImplemented(6)
                    }
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The capabilities of this PAE port.

        ‘suppImplemented’ : A PACP EAP supplicant functions are
            implemented in this PAE if this bit is on.
        ‘authImplemented’ : A PACP EAP authenticator functions are
            implemented in this PAE if this bit is on.

        ‘mkaImplemented’ : The KaY MKA functions are implemented
            in this PAE if this bit is on.

        ‘macsecImplemented’ : The MACsec functions in the
            Controlled Port are implemented in this PAE if this
            bit is on.

        ‘announcementsImplemented’ : The EAPOL announcement can be
            sent in this PAE if this bit is on.

        ‘listenerImplemented’ : This PAE can receive EAPOL announcement
            if this bit is on.

        ‘virtualPortsImplemented’ : Virtual Port functions are
            implemented in this PAE if this bit is on.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 7 }

ieee8021XPaePortVirtualPortsEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Enable or disable to Virtual Ports function for this Real Port
        PAE, the object ieee8021XPaePortType in the same row has the
        value ‘realPort’.  If this PAE is not a Real Port, this object
        should be read only and returns ‘false’.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘virtualPortsImplemented’ off.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XPaePortEntry 8 }

ieee8021XPaePortMaxVirtualPorts OBJECT-TYPE
    SYNTAX          Unsigned32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The maximum number of virtual ports can be supported in this
        port.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 9 }

ieee8021XPaePortCurrentVirtualPorts OBJECT-TYPE
    SYNTAX          Gauge32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The current number of virtual ports is running in this port.”
    REFERENCE       “802.1X Clause 12.9.2, Figure 12-3”
    ::= { ieee8021XPaePortEntry 10 }

ieee8021XPaePortVirtualPortStart OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be ‘true’ if the virtual port is created by
        receipt of an EAPOL-Start packet.”
    REFERENCE       “802.1X Clause 12.7, Figure 12-3”
    ::= { ieee8021XPaePortEntry 11 }

ieee8021XPaePortVirtualPortPeerMAC OBJECT-TYPE
    SYNTAX          MacAddress
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The source MAC address of the received EAPOL-Start if
        ieee8021XPaePortVirtualPortStart is set ‘true’.

        If ieee8021XPaePortVirtualPortStart is not ‘true’ in the same
        row, the value of this object should be 00-00-00-00-00-00.”
    REFERENCE       “802.1X Clause 12.7, Figure 12-3”
    ::= { ieee8021XPaePortEntry 12 }

ieee8021XPaePortLogonEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Enable or disable to transmit network announcement
        information.”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    ::= { ieee8021XPaePortEntry 13 }

ieee8021XPaePortAuthenticatorEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “Enable or disable to the Authenticator function in this PAE.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘authImplemented’ Off.”
    REFERENCE       “802.1X Clause 8.4, Figure 12-3”
    ::= { ieee8021XPaePortEntry 14 }

ieee8021XPaePortSupplicantEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “Enable or disable to the Supplicant function in this PAE.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘suppImplemented’ off.”
    REFERENCE       “802.1X Clause 8.4, Figure 12-3”
    ::= { ieee8021XPaePortEntry 15 }

ieee8021XPaePortKayMkaEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Enable or disable the MKA protocol function in this PAE.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘mkaImplemented’ off.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    ::= { ieee8021XPaePortEntry 16 }

ieee8021XPaePortAnnouncerEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Enable or disable the network Announcer function in this PAE.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘announcementsImplemented’ off.”
    REFERENCE       “802.1X Clause 10.4, Figure 12-3”
    ::= { ieee8021XPaePortEntry 17 }

ieee8021XPaePortListenerEnable OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “Enable or disable the network Listener function in this PAE.

        This object will be read only and returns ‘false’ if the value
        of the object ieee8021XPaePortCapabilities in the same row has
        the bit ‘listenerImplemented’ off.”
    REFERENCE       “802.1X Clause 10.4, Figure 12-3”
    ::= { ieee8021XPaePortEntry 18 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAC Port Table
-- ------------------------------------------------------------------ --

ieee8021XPacPortTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XPacPortEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “A table of system level information for each interface
        supported by PAC.

        This table will be instantiated if the value of the object
        ieee8021XPaePortCapabilities in the corresponding entry of the
        ieee8021XPaePortTable has the bit ‘macsecImplemented’ off.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “IEEE 802.1X Clause 6.4, Clause 14”
    ::= { ieee8021XPaeSystem 6 }

ieee8021XPacPortEntry OBJECT-TYPE
    SYNTAX          Ieee8021XPacPortEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry containing PAC management information applicable to
        a particular interface.”
    INDEX           { ieee8021XPacPortControlledPortNumber }
    ::= { ieee8021XPacPortTable 1 }

Ieee8021XPacPortEntry ::= SEQUENCE {
        ieee8021XPacPortControlledPortNumber    InterfaceIndex,
        ieee8021XPacPortAdminPt2PtMAC           INTEGER,
        ieee8021XPacPortOperPt2PtMAC            TruthValue
}

ieee8021XPacPortControlledPortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “The index to identify the ‘Controlled Port’ interface for a PAC.”
    REFERENCE       “IEEE 802.1X Clause 6.4”
    ::= { ieee8021XPacPortEntry 1 }

ieee8021XPacPortAdminPt2PtMAC OBJECT-TYPE
    SYNTAX          INTEGER  {
                        forceTrue(1),
                        forceFalse(2),
                        auto(3)
                    }
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “An object to control the service connectivity to at most one
        other system.  The ieee8021XPacPortOperPt2PtMAC indicates
        operational status of the service connectivity for this PAC.
        ‘forceTrue’ : allows only one service connection to the
                       other system.

        ‘forceFalse’ : no restriction on the number of service
                        connections to the other systems.

        ‘auto’ : means the service connectivity is determined by the
                  service providing entity.”
    REFERENCE       “IEEE 802.1X Clause 6.4”
    DEFVAL          { auto }
    ::= { ieee8021XPacPortEntry 2 }

ieee8021XPacPortOperPt2PtMAC OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “An object to reflect the current service connectivity status.

        ‘true’ : means the service connectivity of this PAC
             Controlled Port provides at most one other system.

        ‘false’ : means the service connectivity of this PAC could
             provide more than one other system.”
    REFERENCE       “IEEE 802.1X Clause 6.4”
    ::= { ieee8021XPacPortEntry 3 }


-- ---------------------------------------------------------------------- --
-- The 802.1X PAE Logon Process Group
-- ---------------------------------------------------------------------- --
--
-- ---------------------------------------------------------------------- --
-- The 802.1X PAE Logon Process Table
-- ---------------------------------------------------------------------- --

ieee8021XPaePortLogonTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XPaePortLogonEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “A table of system level information for each port to support
        the Logon Process(es) status information.

        This table will be instantiated if the object
        ieee8021XPaePortLogonEnable in the corresponding entry of the
        ieee8021XPaePortTable is ‘true’.”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    ::= { ieee8021XPaeLogon 1 }

ieee8021XPaePortLogonEntry OBJECT-TYPE
    SYNTAX          Ieee8021XPaePortLogonEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry contains Logon Process status information for the
        PAE.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XPaePortLogonTable 1 }

Ieee8021XPaePortLogonEntry ::= SEQUENCE {
        ieee8021XPaePortLogonConnectStatus INTEGER,
        ieee8021XPaePortPortValid          TruthValue
}

ieee8021XPaePortLogonConnectStatus OBJECT-TYPE
    SYNTAX          INTEGER  {
                        pending(1),
                        unauthenticated(2),
                        authenticated(3),
                        secure(4)
                    }
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The Logon Process sets this variable to one of the following
        values, to indicate to the CP state machine if, and how,
        connectivity is to be provided through the Controlled Port :

        'pending' : Prevent connectivity by disabling the
            Controlled Port of this PAE.

        'unauthenticated' : Provide unsecured connectivity, enabling
            the Controlled Port of this PAE.

        'authenticated' : Provide unsecured connectivity but with
            authentication, enabling Controlled Port of this PAE.

        'secure' : Provide secure connectivity, using SAKs provided by
            the KaY (when available) and enabling Controlled Port when
            those keys are installed and in use."
    REFERENCE       "802.1X Clause 12.3, Figure 12-3"
    ::= { ieee8021XPaePortLogonEntry 1 }

ieee8021XPaePortPortValid OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be set 'true' if Controlled Port communication
        is secured as specified by the MACsec."
    REFERENCE       "802.1X Clause 12.3, Figure 12-3"
    ::= { ieee8021XPaePortLogonEntry 2 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE Session Table
-- ------------------------------------------------------------------ --

ieee8021XPaePortSessionTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XPaePortSessionEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "A table of system level information for each port to support
        Logon Process(es) session information.  This table maintains
        session statistics for its associated Controlled Port,
        suitable for communication to a RADIUS or other AAA server at
        the end of a session for accounting purpose.

        This table will be instantiated if the object
        ieee8021XPaePortLogonEnable in the corresponding entry of the
        ieee8021XPaePortTable is 'true'."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaeLogon 2 }

ieee8021XPaePortSessionEntry OBJECT-TYPE
    SYNTAX          Ieee8021XPaePortSessionEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry contains Logon Process session information for the
        PAE.  A session, an entry, begins when the operation of
        Controlled Port becomes 'true' and ends when it becomes
        'false'.

        The counts of frames and octets can be derived from those
        maintained to support from Interface MIB counters for the
        SecY's or the PAC's Controlled Port, but differs in that the
        counts are zeroed when the session begins."
    INDEX           { ieee8021XPaeSessionControlledPortNumber }
    ::= { ieee8021XPaePortSessionTable 1 }

Ieee8021XPaePortSessionEntry ::= SEQUENCE {
        ieee8021XPaeSessionControlledPortNumber   InterfaceIndex,
        ieee8021XPaePortSessionOctetsRx           Counter64,
        ieee8021XPaePortSessionOctetsTx           Counter64,
        ieee8021XPaePortSessionPktsRx             Counter64,
        ieee8021XPaePortSessionPktsTx             Counter64,
        ieee8021XPaePortSessionId                 SnmpAdminString,
        ieee8021XPaePortSessionStartTime          TimeStamp,
        ieee8021XPaePortSessionIntervalTime       TimeInterval,
        ieee8021XPaePortSessionTerminate          INTEGER,
        ieee8021XPaePortSessionUserName           SnmpAdminString
}

ieee8021XPaeSessionControlledPortNumber OBJECT-TYPE
    SYNTAX          InterfaceIndex
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “The index to identify the ‘Controlled Port’ interface’s session
        information for a PAE.”
    REFERENCE       “802.1X Clause 12.5.1, Figure 12-3”
    ::= { ieee8021XPaePortSessionEntry 1 }

ieee8021XPaePortSessionOctetsRx OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           “Octets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of octets received in this session of this PAE.

        Discontinuities in the value of this counter can occur at
        re-initialization of the management system, and at
        other times as indicated by the value of
        ieee8021XPaePortSessionStartTime.”
    REFERENCE       “802.1X Clause 12.5.1, Figure 12-3”
    ::= { ieee8021XPaePortSessionEntry 2 }

ieee8021XPaePortSessionOctetsTx OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           “Octets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of octets transmitted in this session of this PAE.

        Discontinuities in the value of this counter can occur at
        re-initialization of the management system, and at
        other times as indicated by the value of
        ieee8021XPaePortSessionStartTime.”
    REFERENCE       “802.1X Clause 12.5.1, Figure 12-3”
    ::= { ieee8021XPaePortSessionEntry 3 }

ieee8021XPaePortSessionPktsRx OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of packets received in this session of this PAE.

        Discontinuities in the value of this counter can occur at
        re-initialization of the management system, and at
        other times as indicated by the value of
        ieee8021XPaePortSessionStartTime.”
    REFERENCE       “802.1X Clause 12.5.1, Figure 12-3”
    ::= { ieee8021XPaePortSessionEntry 4 }

ieee8021XPaePortSessionPktsTx OBJECT-TYPE
    SYNTAX          Counter64
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of packets transmitted in this session of this PAE.

        Discontinuities in the value of this counter can occur at
        re-initialization of the management system, and at
        other times as indicated by the value of
        ieee8021XPaePortSessionStartTime."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaePortSessionEntry 5 }

ieee8021XPaePortSessionId OBJECT-TYPE
    SYNTAX          SnmpAdminString (SIZE  (3..253))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The session identifier for this session of the PAE.  A UTF-8
        string, uniquely identifying the session within the context of
        the PAE's system."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaePortSessionEntry 6 }

ieee8021XPaePortSessionStartTime OBJECT-TYPE
    SYNTAX          TimeStamp
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The starting time of this session."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaePortSessionEntry 7 }

ieee8021XPaePortSessionIntervalTime OBJECT-TYPE
    SYNTAX          TimeInterval
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The duration time of the session has been last."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaePortSessionEntry 8 }

ieee8021XPaePortSessionTerminate OBJECT-TYPE
    SYNTAX          INTEGER  {
                        macOperFailed(1),
                        sysAccessDisableOrPortInit(2),
                        receiveEapolLogOff(3),
                        eapReauthFailure(4),
                        mkaFailure(5),
                        newSessionBegin(6),
                        notTerminateYet(7)
                    }
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The reason for the session termination, one of the following :

        'macOperFailed' : 'Common Port' for this PAE is not
            operational.

        'sysAccessDisableOrPortInit' : The ieee8021XPaeSysAccessControl
            object is set to 'false' or initialization process of this
            PAE is invoked.

        'receiveEapolLogOff' : The PAE has received EAPOL-Logoff
            frame.

        'eapReauthFailure' : EAP reauthentication has failed.

        'mkaFailure' : MKA failure or other MKA termination.

        'newSessionBegin' : New session beginning.

        'notTerminateYet' : Not Terminated Yet."
    REFERENCE       "802.1X Clause 12.5.1, Figure 12-3"
    ::= { ieee8021XPaePortSessionEntry 9 }

ieee8021XPaePortSessionUserName OBJECT-TYPE
    SYNTAX          SnmpAdminString (SIZE  (0..253))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The session user name for this session in the PAE.  A UTF-8
        string, representing the identity of the peer Supplicant.

        If no such information, zero length string will return.”
    REFERENCE       “802.1X Clause 12.5.1, Figure 12-3”
    ::= { ieee8021XPaePortSessionEntry 10 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE Logon Process NID Table
-- ------------------------------------------------------------------ --

ieee8021XLogonNIDTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XLogonNIDEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “The Logon Process may use Network Identities (NIDs) to manage
        its use of authentication credentials, cached CAKs, and
        announcements.  This table provides the NID information for
        Logon Process.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    ::= { ieee8021XPaeLogon 3 }

ieee8021XLogonNIDEntry OBJECT-TYPE
    SYNTAX          Ieee8021XLogonNIDEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry provides the NID information for a Logon Process.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XLogonNIDTable 1 }

Ieee8021XLogonNIDEntry ::= SEQUENCE {
        ieee8021XLogonNIDConnectedNID Ieee8021XPaeNID,
        ieee8021XLogonNIDRequestedNID Ieee8021XPaeNIDOrNull,
        ieee8021XLogonNIDSelectedNID  Ieee8021XPaeNIDOrNull
}

ieee8021XLogonNIDConnectedNID OBJECT-TYPE
    SYNTAX          Ieee8021XPaeNID
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The NID associated with the current connectivity (possibly
        unauthenticated) provided by the operation of the CP state
        machine.

        This object can differ from both the ieee8021XLogonNIDSelectedNID and
        the ieee8021XLogonNIDRequestedNID objects in the same row if
        authenticated connectivity (either secure or unsecured) has
        already been established, and EAP authentication and MKA
        operation for both of the latter have not met the necessary
        conditions (as specified by the control variables unauthAllowed
        and unsecureAllowed).”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    ::= { ieee8021XLogonNIDEntry 1 }

ieee8021XLogonNIDRequestedNID OBJECT-TYPE
    SYNTAX          Ieee8021XPaeNIDOrNull
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The NID marked as access requested in announcements, as
        determined from EAPOL-Start frames.  The default of this object
        is as the configured value of object ieee8021XLogonNIDSelectedNID.

        This object information provides context for the PAE’s EAP
        Authenticator.  If no EAPOL-Start frame has been received since
        the PAE’s ‘Common Port’ became operational, or the last
        EAPOL-Start frame received for the port did not contain a
        requested NID, the object will take on the value of the object
        ieee8021XLogonNIDSelectedNID in the same row.”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    ::= { ieee8021XLogonNIDEntry 2 }

ieee8021XLogonNIDSelectedNID OBJECT-TYPE
    SYNTAX          Ieee8021XPaeNIDOrNull
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “The NID currently configured for use by an access ‘Controlled
        Port’ when transmitting EAPOL-Start frames.  The default of
        this object is empty string.

        This object may be either explicitly configured by management
        or determined by the PAE using NID selection algorithms.  If no
        authentication is in progress, and the current connectivity is
        terminated and then starts again, ieee8021XLogonNIDConnectedNID will
        take on the value of ieee8021XLogonNIDRequestedNID (though a PAE
        NID’s election algorithm, if used, can subsequently select
        another NID).”
    REFERENCE       “802.1X Clause 12.5, Figure 12-3”
    DEFVAL          { “” }
    ::= { ieee8021XLogonNIDEntry 3 }


-- ------------------------------------------------------------------ --
-- The PAE Authenticator Group
-- ------------------------------------------------------------------ --
--
-- ------------------------------------------------------------------ --
-- The 802.1X PAE Authenticator Table
-- ------------------------------------------------------------------ --

ieee8021XAuthenticatorTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XAuthenticatorEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “A table that contains the configuration objects for the
        Authenticator PAE associated with each port.  This table will
        be instantiated if the object ieee8021XPaePortAuthenticatorEnable in
        the corresponding entry of the ieee8021XPaePortTable is ‘true’.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “802.1X Clause 8, Figure 12-3”
    ::= { ieee8021XPaeAuthenticator 1 }

ieee8021XAuthenticatorEntry OBJECT-TYPE
    SYNTAX          Ieee8021XAuthenticatorEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry that contains the Authenticator configuration objects
        for the PAE.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XAuthenticatorTable 1 }
```

```
Ieee8021XAuthenticatorEntry ::= SEQUENCE {
        ieee8021XAuthPaeAuthenticate  TruthValue,
        ieee8021XAuthPaeAuthenticated TruthValue,
        ieee8021XAuthPaeFailed        TruthValue,
        ieee8021XAuthPaeReAuthEnabled TruthValue,
        ieee8021XAuthPaeQuietPeriod   Unsigned32,
        ieee8021XAuthPaeReauthPeriod  Unsigned32,
        ieee8021XAuthPaeRetryMax      Unsigned32,
        ieee8021XAuthPaeRetryCount    Gauge32
}

ieee8021XAuthPaeAuthenticate OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by the PAE authenticator to
        request authentication, and if this object is ‘true’,
        reauthentication is allowed.

        This object will be ‘false’ while the PAE authenticator revokes
        authentication.”
    REFERENCE       “IEEE 802.1X Clause 8, Figure 12-3”
    ::= { ieee8021XAuthenticatorEntry 1 }

ieee8021XAuthPaeAuthenticated OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by PACP if the PAE authenticator
        currently authenticated, and ‘false’ if the authentication
        fails or is revoked.”
    REFERENCE       “IEEE 802.1X Clause 8, Figure 12-3”
    ::= { ieee8021XAuthenticatorEntry 2 }

ieee8021XAuthPaeFailed OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by PACP if the authentication
        has failed or has been terminated.  The cause could be a
        failure returned by EAP, either immediately or following a
        reauthentication, an excessive number of attempts to
        authenticate (either immediately or upon reauthentication), or
        the authenticator deasserting authenticate, the object
        authPaeAuthenticate in the same row is ‘false’.  The PACP
        will set the object authPaeAuthenticated false as well as
        setting the object ‘true’.”
    REFERENCE       “IEEE 802.1X Clause 8, Figure 12-3”
    ::= { ieee8021XAuthenticatorEntry 3 }

ieee8021XAuthPaeReAuthEnabled OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object is set ‘true’ if PACP should initiate
        reauthentication periodically, ‘false’ otherwise .  Reading
        this object always returns ‘false’.”
    REFERENCE       “IEEE 802.1X Clause 8.9, Figure 12-3”
    ::= { ieee8021XAuthenticatorEntry 4 }

ieee8021XAuthPaeQuietPeriod OBJECT-TYPE
    SYNTAX          Unsigned32 (0..65535)
    UNITS           “seconds”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object indicates a waiting period after a failed
        authentication attempt, before another attempt is permitted.”
    REFERENCE       “IEEE 802.1X Clause 8.6, Figure 12-3”
    DEFVAL          { 60 }
    ::= { ieee8021XAuthenticatorEntry 5 }

ieee8021XAuthPaeReauthPeriod OBJECT-TYPE
    SYNTAX          Unsigned32 (0..65535)
    UNITS           “seconds”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object indicates the time period of the reauthentication
        to the supplicant.”
    REFERENCE       “IEEE 802.1X Clause 8.6, Figure 12-3”
    DEFVAL          { 3600 }
    ::= { ieee8021XAuthenticatorEntry 6 }

ieee8021XAuthPaeRetryMax OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           “times”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “The maximum number of authentication attempts before failure is
        reported to the Logon Process, and the authPaeQuietPeriod
        timer imposed before further attempts are permitted.”
    REFERENCE       “IEEE 802.1X Clause 8.9, Figure 12-3”
    DEFVAL          { 2 }
    ::= { ieee8021XAuthenticatorEntry 7 }

ieee8021XAuthPaeRetryCount OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           “times”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The count of the number of authentication attempts.”
    REFERENCE       “IEEE 802.1X Clause 8.9”
    ::= { ieee8021XAuthenticatorEntry 8 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE Supplicant Group
-- ------------------------------------------------------------------ --
--
-- ------------------------------------------------------------------ --
-- The 802.1X PAE Supplicant Table
-- ------------------------------------------------------------------ --

ieee8021XSupplicantTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XSupplicantEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “A table that contains the configuration objects for the
        Supplicant PAE associated with each port.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “802.1X Clause 8, Figure 8-6, Figure 12-3”
    ::= { ieee8021XPaeSupplicant 1 }

ieee8021XSupplicantEntry OBJECT-TYPE
    SYNTAX          Ieee8021XSupplicantEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “The configuration information for an Supplicant PAE.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XSupplicantTable 1 }
```

```
Ieee8021XSupplicantEntry ::= SEQUENCE {
        ieee8021XSuppPaeAuthenticate  TruthValue,
        ieee8021XSuppPaeAuthenticated TruthValue,
        ieee8021XSuppPaeFailed        TruthValue,
        ieee8021XSuppPaeHelloPeriod   Unsigned32,
        ieee8021XSuppPaeRetryMax      Unsigned32,
        ieee8021XSuppPaeRetryCount    Gauge32
}

ieee8021XSuppPaeAuthenticate OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by the PAE supplicant to request
        authentication, and if this object is ‘true’, reauthentication
        is allowed.

        This object will be ‘false’ while the PAE supplicant revokes
        authentication.”
    REFERENCE       “IEEE 802.1X Clause 8.4, Figure 8-6, Figure 12-3”
    ::= { ieee8021XSupplicantEntry 1 }

ieee8021XSuppPaeAuthenticated OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by PACP if the PAE supplicant
        currently authenticated, and ‘false’ if the authentication
        fails or is revoked.”
    REFERENCE       “IEEE 802.1X Clause 8.4, Figure 8-6, Figure 12-3”
    ::= { ieee8021XSupplicantEntry 2 }

ieee8021XSuppPaeFailed OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ by PACP if the authentication
        has failed or has been terminated.  The cause could be a
        failure returned by EAP, either immediately or following a
        reauthentication, an excessive number of attempts to
        authenticate (either immediately or upon reauthentication), or
        the supplicant deasserting authenticate, the object
        ieee8021XSuppPaeAuthenticate in the same row is ‘false’.  The PACP
        will set the object ieee8021XSuppPaeAuthenticated false as well as
        setting the object ‘true’.”
    REFERENCE       “IEEE 802.1X Clause 8.4, Figure 8-6, Figure 12-3”
    ::= { ieee8021XSupplicantEntry 3 }

ieee8021XSuppPaeHelloPeriod OBJECT-TYPE
    SYNTAX          Unsigned32 (0..65535)
    UNITS           “seconds”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object indicated a waiting time period after a failed
        authentication attempt, before another attempt is permitted.”
    REFERENCE       “IEEE 802.1X Clause 8.6, Figure 8-6, Figure 12-3”
    DEFVAL          { 60 }
    ::= { ieee8021XSupplicantEntry 4 }

ieee8021XSuppPaeRetryMax OBJECT-TYPE
    SYNTAX          Unsigned32
    UNITS           “times”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “The maximum number of authentication attempts before failure is
        reported to the Logon Process, and the ieee8021XSuppPaeHelloPeriod
        timer imposed before further attempts are permitted.”
    REFERENCE       “IEEE 802.1X Clause 8.7, Figure 8-6, Figure 12-3”
    DEFVAL          { 2 }
    ::= { ieee8021XSupplicantEntry 5 }

ieee8021XSuppPaeRetryCount OBJECT-TYPE
    SYNTAX          Gauge32
    UNITS           “times”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The count of the number of authentication attempts.”
    REFERENCE       “IEEE 802.1X Clause 8.7, Figure 8-6, Figure 12-3”
    ::= { ieee8021XSupplicantEntry 6 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE EAPOL Statistics Table
-- ------------------------------------------------------------------ --

ieee8021XEapolStatsTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XEapolStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “A table in system level contains the EAPOL statistics and
        diagnostics information supported by PAE.”
    REFERENCE       “802.1X Clause 12.8, Figure 12-3”
    ::= { ieee8021XPaeEapol 1 }

ieee8021XEapolStatsEntry OBJECT-TYPE
    SYNTAX          Ieee8021XEapolStatsEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry contains the EAPOL statistics and diagnostics
        information for a PAE.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XEapolStatsTable 1 }

Ieee8021XEapolStatsEntry ::= SEQUENCE {
        ieee8021XEapolInvalidFramesRx          Counter32,
        ieee8021XEapolEapLengthErrorFramesRx   Counter32,
        ieee8021XEapolAnnouncementFramesRx     Counter32,
        ieee8021XEapolAnnouncementReqFramesRx Counter32,
        ieee8021XEapolPortUnavailableFramesRx Counter32,
        ieee8021XEapolStartFramesRx            Counter32,
        ieee8021XEapolEapFramesRx              Counter32,
        ieee8021XEapolLogoffFramesRx           Counter32,
        ieee8021XEapolMkNoCknFramesRx          Counter32,
        ieee8021XEapolMkInvalidFramesRx        Counter32,
        ieee8021XEapolLastRxFrameVersion       Unsigned32,
        ieee8021XEapolLastRxFrameSource        MacAddress,
        ieee8021XEapolSuppEapFramesTx          Counter32,
        ieee8021XEapolLogoffFramesTx           Counter32,
        ieee8021XEapolAnnouncementFramesTx     Counter32,
        ieee8021XEapolAnnouncementReqFramesTx Counter32,
        ieee8021XEapolStartFramesTx            Counter32,
        ieee8021XEapolAuthEapFramesTx          Counter32,
        ieee8021XEapolMkaFramesTx              Counter32
}

ieee8021XEapolInvalidFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of invalid EAPOL frames of any type that have been
        received by this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 1 }
```

```
ieee8021XEapolEapLengthErrorFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL frames that the Packet Body Length does not
        match a Packet Body that is contained within the octets of the
        received EAPOL MPDU in this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 2 }

ieee8021XEapolAnnouncementFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL-Announcement frames that have been received
        by this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 3 }

ieee8021XEapolAnnouncementReqFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL-Announcement-Req frames that have been
        received by this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 4 }

ieee8021XEapolPortUnavailableFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL frames that are discarded because their
        processing would require the creation of a virtual port, for
        which there are inadequate or constrained resources, or an
        existing virtual port and no such port currently exists.  If
        virtual port is not supported, this object should be always 0.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 5 }

ieee8021XEapolStartFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL-Start frames that have been received by
        this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 6 }

ieee8021XEapolEapFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           “Packets”
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The number of EAPOL-EAP frames that have been received by
        this PAE.”
    REFERENCE       “802.1X Clause 12.8.1, Figure 12-3”
    ::= { ieee8021XEapolStatsEntry 7 }

ieee8021XEapolLogoffFramesRx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS               "Packets"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The number of EAPOL-Logoff frames that have been received by
        this PAE."
    REFERENCE           "802.1X Clause 12.8.1, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 8 }

ieee8021XEapolMkNoCknFramesRx OBJECT-TYPE
    SYNTAX              Counter32
    UNITS               "Packets"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The number of MKPDUs received with MKA not enabled or CKN not
        recognized in this PAE."
    REFERENCE           "802.1X Clause 12.8.1, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 9 }

ieee8021XEapolMkInvalidFramesRx OBJECT-TYPE
    SYNTAX              Counter32
    UNITS               "Packets"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The number of MKPDUs failing in message authentication on
        receipt process in this PAE."
    REFERENCE           "802.1X Clause 12.8.1, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 10 }

ieee8021XEapolLastRxFrameVersion OBJECT-TYPE
    SYNTAX              Unsigned32
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The version of last received EAPOL frame by this PAE."
    REFERENCE           "802.1X Clause 12.8.2, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 11 }

ieee8021XEapolLastRxFrameSource OBJECT-TYPE
    SYNTAX              MacAddress
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The source MAC address of last received EAPOL frame by this
        PAE."
    REFERENCE           "802.1X Clause 12.8.2, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 12 }

ieee8021XEapolSuppEapFramesTx OBJECT-TYPE
    SYNTAX              Counter32
    UNITS               "Packets"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The number of EAPOL-EAP frames that have been transmitted by
        the supplicant of this PAE."
    REFERENCE           "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 13 }

ieee8021XEapolLogoffFramesTx OBJECT-TYPE
    SYNTAX              Counter32
    UNITS               "Packets"
    MAX-ACCESS          read-only
    STATUS              current
    DESCRIPTION
        "The number of EAPOL-Logoff frames that have been transmitted by
        this PAE."
    REFERENCE           "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 14 }
```

```
ieee8021XEapolAnnouncementFramesTx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           "Packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of EAPOL-Announcement frames that have been
        transmitted by this PAE."
    REFERENCE       "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 15 }

ieee8021XEapolAnnouncementReqFramesTx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           "Packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of EAPOL-Announcement-Req frames that have been
        transmitted by this PAE."
    REFERENCE       "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 16 }

ieee8021XEapolStartFramesTx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           "Packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of EAPOL-Start frames that have been received by
        this PAE."
    REFERENCE       "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 17 }

ieee8021XEapolAuthEapFramesTx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           "Packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of EAPOL-EAP frames that have been transmitted by
        the authenticator of this PAE."
    REFERENCE       "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 18 }

ieee8021XEapolMkaFramesTx OBJECT-TYPE
    SYNTAX          Counter32
    UNITS           "Packets"
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The number of EAPOL-MKA frames with no CKN information that
        have been transmitted by this PAE."
    REFERENCE       "802.1X Clause 12.8.3, Figure 12-3"
    ::= { ieee8021XEapolStatsEntry 19 }


-- ------------------------------------------------------------------ --
-- The 802.1X PAE KaY Group
-- ------------------------------------------------------------------ --
--
-- ------------------------------------------------------------------ --
-- The 802.1X PAE KaY Table
-- ------------------------------------------------------------------ --

ieee8021XKayMkaTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XKayMkaEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "A table of system level information for each interface
        supported by the KaY (Key Agreement Entity).  This table will
        be instantiated if the object ieee8021XPaePortKayMkaEnable in
        the corresponding entry of the ieee8021XPaePortTable is ‘true’.

        The following terms are used to identify roles within the MKA
        protocol or protocol scenarios and the MIB description :

        participant : An instance of MKA, transmitting and receiving
            frames protected by keys derived from a single CAK, and
            operating with positive intent, obeying the protocol.

        member: A participant that possesses the CAK that can be used
            to prove liveness and to obtain membership in the CA under
            discussion.

        actor: The participant under discussion, usually in the KaY
            being described.

        partners: Participants or members attached to the same LAN as
            the actor, excluding the actor.

        principal actor: The actor controlling the PAC or SecY
            associated with the KaY.

        Each participant selects the live participant advertising the
        highest priority as its key server provided that participant
        has not selected another as its key server or is unwilling to
        act as the key server.  If a key server cannot be selected SAKs
        are not distributed.  In the event of a tie for highest
        priority key server, the member with the highest priority SCI
        is chosen.  For consistency with other uses of the SCI’s MAC
        Address component as a priority, numerically lower values of
        the key server priority and SCI are accorded the highest
        priority.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    ::= { ieee8021XPaeKaY 1 }

ieee8021XKayMkaEntry OBJECT-TYPE
    SYNTAX          Ieee8021XKayMkaEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        “An entry containing KaY MKA management information applicable
        to a particular interface.”
    INDEX           { ieee8021XPaePortNumber }
    ::= { ieee8021XKayMkaTable 1 }

Ieee8021XKayMkaEntry ::= SEQUENCE {
        ieee8021XKayMkaActive
            TruthValue,
        ieee8021XKayMkaAuthenticated
            TruthValue,
        ieee8021XKayMkaSecured
            TruthValue,
        ieee8021XKayMkaFailed
            TruthValue,
        ieee8021XKayMkaActorSCI
            SecySCI,
        ieee8021XKayMkaActorsPriority
            Ieee8021XMkaKeyServerPriority,
        ieee8021XKayMkaKeyServerPriority
            Ieee8021XMkaKeyServerPriority,
        ieee8021XKayMkaKeyServerSCI
            SecySCI,
        ieee8021XKayAllowedJoinGroup
            TruthValue,
        ieee8021XKayAllowedFormGroup
            TruthValue,
        ieee8021XKayCreateNewGroup
```

```
            TruthValue,
        ieee8021XKayMacSecCapability
            INTEGER,
        ieee8021XKayMacSecDesired
            TruthValue,
        ieee8021XKayMacSecProtect
            TruthValue,
        ieee8021XKayMacSecReplayProtect
            TruthValue,
        ieee8021XKayMacSecValidate
            TruthValue,
        ieee8021XKayMacSecConfidentialityOffset
            Integer32,
        ieee8021XKayMkaTxKN
            Ieee8021XMkaKN,
        ieee8021XKayMkaTxAN
            RowPointer,
        ieee8021XKayMkaRxKN
            Ieee8021XMkaKN,
        ieee8021XKayMkaRxAN
            RowPointer,
        ieee8021XKayMkaSuspendFor
            INTEGER,
        ieee8021XKayMkaSuspendOnRequest
            TruthValue,
        ieee8021XKayMkaSuspendedWhile
            INTEGER
}

ieee8021XKayMkaActive OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if there is at least one MKA active
        actor, transmitting MKPDUs"
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 1 }

ieee8021XKayMkaAuthenticated OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if the principal actor,
        i.e. the actor controlling the PAC or SecY associated with
        the KaY, has determined that Controlled Port communication
        communication should proceed without MACsec."
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 2 }

ieee8021XKayMkaSecured OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if the principal actor has
        determined that communication should use MACsec."
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 3 }

ieee8021XKayMkaFailed OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if the object
        ieee8021XKayMkaSecured in
        the same row is 'false' and MKA Life Time has elapsed since an
        MKA participant was last created."
    REFERENCE       "IEEE 802.1X Clause 9.16, Table 9-3, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 4 }
```

```
ieee8021XKayMkaActorSCI OBJECT-TYPE
    SYNTAX          SecySCI
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The SCI assigned by the system to the port, applies to all the
        port's MKA actors."
    REFERENCE
        "IEEE 802.1X Clause 9.16, Figure 12-3
         IEEE 802.1AE Clause 7.1.2, 10.7.1"
    ::= { ieee8021XKayMkaEntry 5 }

ieee8021XKayMkaActorsPriority OBJECT-TYPE
    SYNTAX          Ieee8021XMkaKeyServerPriority
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        "The Key Server priority for all the port's MKA actors.  Each
        participant encodes a key server priority, an 8-bit integer, in
        each MKPDU."
    REFERENCE       "IEEE 802.1X Clause 9.16, Table 9-2, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 6 }

ieee8021XKayMkaKeyServerPriority OBJECT-TYPE
    SYNTAX          Ieee8021XMkaKeyServerPriority
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The priority of the elected Key Server through MKA in the CA."
    REFERENCE       "IEEE 802.1X Clause 9.16, Table 9-2, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 7 }

ieee8021XKayMkaKeyServerSCI OBJECT-TYPE
    SYNTAX          SecySCI
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The SCI for key server for the MKA principal actor.  The length
        of this object is 0 if there is no principal actor, or that
        actor has no live peers.  This object matches the
        ieee8021XKayMkaActorSCI object in the same row if the actor is
        the key server."
    REFERENCE
        "IEEE 802.1X Clause 9.16, Figure 12-3
         IEEE 802.1AE Clause 7.1.2, 10.7.1"
    ::= { ieee8021XKayMkaEntry 8 }

ieee8021XKayAllowedJoinGroup OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if the KaY will accept Group CAKs
        distributed by MKA protocol."
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 9 }

ieee8021XKayAllowedFormGroup OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object will be 'true' if the KaY will attempt to use
        point-to-point CAKs to distribute a group CAK, if it is the
        Key Server for the MKA instances for all the point-to-point CAKs."
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 10 }

ieee8021XKayCreateNewGroup OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object is set ‘true’ if a new Group CAK is to be
        distributed if the KaY is the Key Server for the MKA instances
        for all the point-to-point CAKs.  This object will be set ‘false’
        by the KaY when distribution is complete.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaEntry 11 }

ieee8021XKayMacSecCapability OBJECT-TYPE
    SYNTAX          INTEGER {
                        noMACsec(0),
                        macSecCapability1(1),
                        macSecCapability2(2),
                        macSecCapability3(3)
                    }
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object indicates whether MACsec is implemented, and if so
        whether the implementation provides integrity protection only,
        integrity and integrity with confidentiality, or integrity and
        integrity with confidentiality with a selectable confidentiality offset
        of 0, 30, or 50 octets (see IEEE Std 802.1AE).

        ‘noMACsec’ : the MACsec is not implemented.

        ‘macSecCapability1’ :  capable in ‘integrity protection without
             confidentiality’.

        ‘macSecCapability2’ :  capable in ‘integrity protection without
             confidentiality’ and integrity protection and confidentiali
              with a confidentiality offset 0’.

        ‘macSecCapability3’ :  capable in ‘integrity protection without
             confidentiality’ and integrity protection and confidentiali
              with a confidentiality offset 0, 30 or 50’.”
    REFERENCE
        “IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-3, Table 11-6”
    ::= { ieee8021XKayMkaEntry 12 }

ieee8021XKayMacSecDesired OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “This object will be set ‘true’ if the MKA participants desire
        the use of MACsec to protect frames with this KaY.”
    REFERENCE
        “IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaEntry 13 }

ieee8021XKayMacSecProtect OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The status of the MACsec protection function for this KaY.

        ‘true’ : then the status of the MACsec protection function will
            be as object secyIfProtectFramesEnable object configured
            in the IEEE8021-SECY-MIB.
        ‘false’ : then the MACsec protection function is disabled by
              this KaY.”
    REFERENCE
        “IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-2,
         Figure 12-3, IEEE 802.1AE IEEE8021-SECY-MIB”
    ::= { ieee8021XKayMkaEntry 14 }

ieee8021XKayMacSecReplayProtect OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The status of the MACsec replay protection function for this
        KaY.

        ‘true’ : then the status of the MACsec replay protection
            function will be as secyIfReplayProtectEnable object
            configured in the IEEE8021-SECY-MIB.
        ‘false’ : then the MACsec replay protection function is
            disabled by this KaY.”
    REFERENCE
        “IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-2,
         Figure 12-3”
    ::= { ieee8021XKayMkaEntry 15 }

ieee8021XKayMacSecValidate OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The status of the MACsec validation function for this KaY.

        ‘true’ : then the status of the MACsec validation function
            will be as secyIfValidateFrames object configured in the
            IEEE8021-SECY-MIB.
        ‘false’ : then the MACsec validation function is enabled but
            only for checking without filtering out invalid frames by
            the SecY.”
    REFERENCE
        “IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-2,
         Figure 12-3”
    ::= { ieee8021XKayMkaEntry 16 }

ieee8021XKayMacSecConfidentialityOffset OBJECT-TYPE
    SYNTAX          Integer32 (0 | 30 | 50)
    UNITS           “bytes”
    MAX-ACCESS      read-write
    STATUS          current
    DESCRIPTION
        “The confidentiality protection offset options for the selected
        cipher suite in the MACsec.  If the cipher suite does not have
        this capability, the configured value of the object will not
        apply to the cipher suite.”
    REFERENCE
        “IEEE 802.1X Clause 9.7.1, Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaEntry 17 }

ieee8021XKayMkaTxKN OBJECT-TYPE
    SYNTAX          Ieee8021XMkaKN
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The key number assigned by the key server to the SAK currently
        being used for transmission.  This object will be 0 if MACsec
        is not being used or the key number is not available yet.”
    REFERENCE       “IEEE 802.1X Clause 9.8, Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaEntry 18 }

ieee8021XKayMkaTxAN OBJECT-TYPE
    SYNTAX          RowPointer
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “The AN assigned by the key server for use with the key number
        for transmission.

        This row pointer will point to an entry in the secyTxSATable
        which the secyTxSCEncodingSA object also points to in the
        IEEE8021-SECY-MIB.

        If MACsec is not in use or the AN is not identified yet, the
        value of this object shall be set to the OBJECT IDENTIFIER
        { 0 0 }."
    REFERENCE
        "IEEE 802.1X Clause 9.9, Clause 9.16, Figure 12-3,
         IEEE8021-SECY-MIB"
    ::= { ieee8021XKayMkaEntry 19 }

ieee8021XKayMkaRxKN OBJECT-TYPE
    SYNTAX          Ieee8021XMkaKN
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The key number assigned by the key server to the oldest SAK
        currently being used for reception.  It is the same as the key
        number for transmission if a single SAK is currently in use.
        This object will be 0 if MACsec is not being used or the key
        number is not available yet."
    REFERENCE       "IEEE 802.1X Clause 9.8, Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 20 }

ieee8021XKayMkaRxAN OBJECT-TYPE
    SYNTAX          RowPointer
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The AN assigned by the key server for use with the key number
        for reception.  It is the same as AN for transmission if a
        single SAK is currently in use.

        This row pointer will point to an entry in the secyRxSATable
        which the secyRxSCCurrentSA object also points to in the
        IEEE8021-SECY-MIB.

        If MACsec is not in use or the AN is not identified yet, the
        value of this object shall be set to the OBJECT IDENTIFIER
        { 0 0 }."
    REFERENCE
        "IEEE 802.1X Clause 9.6.1, Clause 9.16, Figure 12-3,
         IEEE8021-SECY-MIB"
    ::= { ieee8021XKayMkaEntry 21 }

ieee8021XKayMkaSuspendFor OBJECT-TYPE
    SYNTAX INTEGER (1..120)
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "Set by management to a non-zero number of seconds between 1
        and MKA Suspension Limit to initiate a suspension (9.18) of
        that duration (if the KaY's principal actor is the Key
        Server) or to request a suspension (otherwise)"
    REFERENCE "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 22 }

ieee8021XKayMkaSuspendOnRequest OBJECT-TYPE
    SYNTAX TruthValue
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
        "The status of the suspendOnRequest function for this KaY.
        'true' : then the KaY's principal actor will initiate a
        suspension if it is the Key Server and another participant
        has requested a suspension by transmitting a non-zero value
        of its suspendFor parameter
        'false' : then the KaY will not initiate a suspension on
        request from another participant."
    REFERENCE "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 23 }

ieee8021XKayMkaSuspendedWhile OBJECT-TYPE
    SYNTAX INTEGER (1..126)
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
```

```
        "Read by management to determine if a suspension is in
        progress and to discover the remaining duration of that
        suspension. May be set directly to coordinate in-service
        upgrades."
    REFERENCE "IEEE 802.1X Clause 5.11.4, Clause 9.16, Clause 9.18.5,
               Clause 9.18.6, Figure 12-3"
    ::= { ieee8021XKayMkaEntry 24 }

-- ------------------------------------------------------------------ --
-- The 802.1X PAE KaY MKA Participants Table
-- ------------------------------------------------------------------ --

ieee8021XKayMkaParticipantTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF Ieee8021XKayMkaParticipantEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "A table for each MKA participant supported by the KaY MKA
        entity.

        For the writeable objects in this table, the configured value
        shall be stored in persistent memory and remain unchanged
        across a re-initialization of the management system of the
        entity."
    REFERENCE       "IEEE 802.1X Clause 9.14, Clause 9.16, Figure 12-3"
    ::= { ieee8021XPaeKaY 2 }

ieee8021XKayMkaParticipantEntry OBJECT-TYPE
    SYNTAX          Ieee8021XKayMkaParticipantEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry containing KaY MKA management information applicable
        to a MKA participant."
    INDEX           { ieee8021XPaePortNumber, ieee8021XKayMkaPartCKN }
    ::= { ieee8021XKayMkaParticipantTable 1 }

Ieee8021XKayMkaParticipantEntry ::= SEQUENCE {
        ieee8021XKayMkaPartCKN                Ieee8021XPaeCKN,
        ieee8021XKayMkaPartKMD                Ieee8021XPaeKMD,
        ieee8021XKayMkaPartNID                Ieee8021XPaeNID,
        ieee8021XKayMkaPartCached             TruthValue,
        ieee8021XKayMkaPartActive             TruthValue,
        ieee8021XKayMkaPartRetain             TruthValue,
        ieee8021XKayMkaPartActivateControl INTEGER,
        ieee8021XKayMkaPartPrincipal          TruthValue,
        ieee8021XKayMkaPartDistCKN            Ieee8021XPaeCKNOrNull,
        ieee8021XKayMkaPartRowStatus          RowStatus
}

ieee8021XKayMkaPartCKN OBJECT-TYPE
    SYNTAX          Ieee8021XPaeCKN
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "The CKN information for this MKA participant."
    REFERENCE       "IEEE 802.1X Clause 9.16, Figure 12-3"
    ::= { ieee8021XKayMkaParticipantEntry 1 }

ieee8021XKayMkaPartKMD OBJECT-TYPE
    SYNTAX          Ieee8021XPaeKMD
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
        "The KMD information for this MKA participant."
    REFERENCE       "IEEE 802.1X Clause 9.16, Clause 12.6, Figure 12-3"
    ::= { ieee8021XKayMkaParticipantEntry 2 }

ieee8021XKayMkaPartNID OBJECT-TYPE
    SYNTAX          Ieee8021XPaeNID
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
        “The NID information for this MKA participant.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Clause 12.6, Figure 12-3”
    ::= { ieee8021XKayMkaParticipantEntry 3 }

ieee8021XKayMkaPartCached OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
        “This object is set ‘true’ by the KaY if the participant’s
        parameters are cached.  If this object is ‘true’, this object
        can be set ‘false’ cleared by management to remove the
        participant’s parameters from the cache.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaParticipantEntry 4 }

ieee8021XKayMkaPartActive OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        “This object is set ‘true’ if the participant is active, i.e. is
        currently transmitting periodic MKPDUs.”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    DEFVAL { false }
    ::= { ieee8021XKayMkaParticipantEntry 5 }

ieee8021XKayMkaPartRetain OBJECT-TYPE
    SYNTAX          TruthValue
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
        “This object is set ‘true’ to retain the participant in the
        cache, even if the KaY would normally remove it (due to lack
        of use for example)”
    REFERENCE       “IEEE 802.1X Clause 9.16, Figure 12-3”
    ::= { ieee8021XKayMkaParticipantEntry 6 }

ieee8021XKayMkaPartActivateControl OBJECT-TYPE
    SYNTAX          INTEGER  {
                        default(1),
                        disabled(2),
                        onOperUp(3),
                        always(4)
                    }
    MAX-ACCESS      read-create
    STATUS          current
    DESCRIPTION
        “This object is for controlling the participant’s behavior when
        the participant is activated.

        ‘default’ : the participant is from cached entries created by
            the KaY as part of normal operation, without explicit
            management, and is activated according to the
            implementation dependent policies of the KaY.

        ‘disabled’ : the participant allows the cache information to
            be retained, but disabled for indefinite period.

        ‘onOperUp’ : causing the participant to be activated when the
            PAE’s ‘Uncontrolled Port’ becomes operational and when the
            PAE resumes following suspension.

        ‘always’ : causing the participant to remain active all the
            time, even in the continued absence of partners.

        If the object changed to disabled(1) or onOperUp(3), the
        participant ceases operation immediately and receipt of MKPDUs
        with a matching CKN during a subsequent period of twice MKA
        lifetime will not cause the participant to become active once
        more.”
```