Telecommunications and Information Exchange Between Systems

ISO/IEC JTC 1/SC 6

Document Number:	6N14123
Date:	2009-10-29
Replaces:	
Document Type:	Text for NP ballot
Document Title:	Text for NP ballot, Information technology —Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part XX: Alternative security mechanism for use with ISO/IEC 8802-11
Document Source:	National Body of China
Project Number:	
Document Status:	SC 6 NBs are requested to ballot through the ISO e-balloting system (www.iso.org/jtc1/sc6) no later than 2010-01-29.
Action ID:	LB
Due Date:	2010-01-29
No. of Pages:	169

ISO/IEC JTC1/SC6 Secretariat Ms. Jooran Lee, KSA (on behalf of KATS)

Korea Technology Center #701-7 Yeoksam-dong, Gangnam-gu, Seoul, 135-513, Republic of Korea ;

Telephone: +82 2 6009 4808 ; Facsimile: +82 2 6009 4819 ; Email : jooran@kisi.or.kr

February 2004

PROPOSAL FOR A NEW WORK ITEM

Date of presentation of proposal: 2009-10-26	Proposer: SAC, China NB
Secretariat: National Body: KATS, Korea NB	**ISO/IEC JTC 1 N** ISO/IEC JTC 1/SC6 N 14123

A proposal for a new work item shall be submitted to the secretariat of the ISO/IEC joint technical committee concerned with a copy to the ISO Central Secretariat.

Presentation of the proposal - to be completed by the proposer. .

Title (subject to be covered and type of standard, e.g. terminology, method of test, performance requirements, etc.)

Information technology—Telecommunications and Information Exchange Between Systems — Local and metropolitan area networks — Specific requirements — Part XX: Alternative Security Mechanism for use with ISO/IEC 8802-11

Scope (and field of application)

ISO/IEC 8802 is a family of standards for local and metropolitan area networks. Specifically, in this part of ISO/IEC 8802, a security mechanism, WLAN Authentication and Privacy Infrastructure (WAPI) is defined.

The purpose of this part of ISO/IEC 8802 is to provide an alternative security mechanism to for use with ISO/IEC 8802-11, for secure wireless connectivity to automatic machinery, equipment or stations that require rapid deployment, which may be portable or hand-held, or which may be mounted on moving vehicles within a local area. For the coexistence of different security mechanisms, this part of ISO/IEC 8802 mainly provides an enhanced security mechanism (WAPI), which is different from those of ISO/IEC 8802-11:2005, Amd 4:2006, Amd 5:2006 and Amd 6:2006.

Purpose and justification - attach a separate page as annex, if necessary

It is a well known fact that current WLAN international standards contains serious security loopholes which need to be dealt with by enhanced security mechanisms. For example:

- Issue 1: 2007-11, *WLAN Epidemiology: Can Your Neighbors' Router Make Yours Sick?*
  - ✓ In densely populated urban areas WLAN routers form a tightly interconnected proximity network that can be exploited as a substrate for the spreading of malware able to launch massive fraudulent attack and affect entire urban areas WLAN networks. In this paper we consider several scenarios for the deployment of malware that spreads solely over the wireless channel of major urban areas in the US.
  - ✓ We uncover a major weakness of WLAN networks in that most of the simulated scenarios show tens of thousands of routers infected in as little time as two weeks, with the majority of the infections occurring in the first 24 to 48 hours.
- Issue 2: 2008-01, *A Wi-Fi virus outbreak? Researchers say it's possible (http://www.networkworld.com/news/2008/010408-a-wi-fi-virus-outbreak-researchers.html)*
  - ✓ If criminals were to target unsecured wireless routers, they could create an attack that could piggyback across thousands of WLAN networks in urban areas, according to researchers.
  - ✓ Even some routers that use encryption could be cracked, if they use the popular WEP (Wired Equivalent Privacy) algorithm, which security experts can crack in just a few minutes now.
- Issue 3: 2008-11, *Attack on WPA*
  - ✓ Beck-Tews attack (Germany 2008.11) from "Practical attacks against WEP and WPA" by Martin Beck and TU-Dresden.
    The attack targets are only WPA implementations those support IEEE802.11e QoS features.
  - ✓ Ohigashi-Morii attack (Japan 2009.8) from "A Practical Message Falsication Attack on WPA" by Toshihiro Ohigashi1 and Masakatu Morii.
    The attack targets are any WPA implementation.

In order to provide a security enhancement on current WLAN:

- Resolution 6.1.10--Encouragement of Study on WLAN Security Work (ISO/IEC JTC 1/SC 6 Plenary Meeting,11 April 2008, Geneva, Switzerland, 6N13602)
  SC 6 encourages representatives from China, IEEE-SA, other SC 6 National Bodies, and other interested parties, to meet to prepare a plan to incorporate WAPI into ISO/IEC 8802-11 within the framework of an SC 6 study period as defined in 12.3 of the JTC1 Directives.
- Resolution 6.1.4--Resolution on WLAN Security (ISO/IEC JTC 1/SC 6 Plenary Meeting, 5 June 2009,Tokyo, Japan, 6N14026)
  SC 6 encourages and welcomes the Chinese National Body to submit the WAPI technology specification for adoption as a stand-alone standard to be used as an alternative security mechanism with ISO/IEC 8802-11standards.

So, this new work item proposal is proposed.

Programme of work

If the proposed new work item is approved, which of the following document(s) is (are) expected to be developed?

__X_ a single International Standard

___ more than one International Standard (expected number: .....)

_____ a multi-part International Standard consisting of ..... parts

_____ an amendment or amendments to the following International Standard(s) .....

_____ a technical report , type .....

And which standard development track is recommended for the approved new work item?

_ X _a. Default Timeframe

____b. Accelerated Timeframe

____c. Extended Timeframe

(note: we propose to have a default timeframe but may convert to accelerated timeframe at any time in the process if it is deemed suitable).

Relevant documents to be considered

ISO/IEC 8802-11 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.

6N13602, Resolutions of ISO/IEC JTC 1/SC 6 Plenary Meeting 11 April 2008 Geneva, Switzerland
6N13719, Meeting minutes of ISO/IEC JTC1/SC6 WAPI Special Meeting (July 22-23, Xi'an, China)

6N13774, WAPI Independent IS Sample Draft

6N13776, The Validity of WAPI IS Option

6N13810, Resolutions of ISO/IEC JTC 1/SC 6 Plenary Meeting, 7 November 2008, Montreux, Switzerland

6N14026, Resolutions of ISO/IEC JTC 1/SC 6 Plenary Meeting, 5 June 2009,Tokyo, Japan

Co-operation and liaison

Preparatory work offered with target date(s)

The preparatory work has been done. A draft of the standard has been prepared and submitted along with this form for consideration.

Signature:

Will the service of a maintenance agency or registration authority be required? .. No.....
- If yes, have you identified a potential candidate? .....
- If yes, indicate name .....

Are there any known requirements for coding? ... No.....
-If yes, please specify on a separate page

Does the proposed standard concern known patented items? .... Yes.....
- If yes, please provide full information in an annex

Are there any known accessibility requirements and/or dependencies (see:

http://www.jtc1access.org)?... No....

-If yes, please specify on a separate page

Are there any known requirements for cultural and linguistic adaptability?... No.....
-If yes, please specify on a separate page

Comments and recommendations of the JTC 1 or SC XXSecretariat - attach a separate page as an annex, if necessary

Comments with respect to the proposal in general, and recommendations thereon: It is proposed to assign this new item to JTC 1/SC 6

Voting on the proposal - Each P-member of the ISO/IEC joint technical committee has an obligation to vote within the time limits laid down (normally three months after the date of circulation).

Date of circulation: 2009-10-29	**Closing date for voting:** 2010-01-29	**Signature of Secretary:** **Ms. Jooran Lee**

NEW WORK ITEM PROPOSAL - PROJECT ACCEPTANCE CRITERIA		
Criterion	**Validity**	**Explanation**
A. Business Requirement		
A.1 Market Requirement	Essential _ X __ Desirable ___ Supportive ___	
A.2 Regulatory Context	Essential _ X __ Desirable ___ Supportive ___ Not Relevant ___	
B. Related Work		
B.1 Completion/Maintenance of current standards	Yes ___ No_ X __	
B.2 Commitment to other organisation	Yes ___ No_ X __	
B.3 Other Source of standards	Yes ___ No_ X __	
C. Technical Status		
C.1 Mature Technology	Yes _ X _ No___	
C.2 Prospective Technology	Yes ___ No_ X __	
C.3 Models/Tools	Yes ___ No_ X __	

D. Conformity Assessment and Interoperability		
D.1 Conformity Assessment	Yes ___ No_ X __	
D.2 Interoperability	Yes ___ No_ X __	
E. Cultural and Linguistic Adaptability		
E.1 Cultural and Linguistic Adaptability	Yes _ __ No_X__	
E.2 Adaptability to Human Functioning and Context of Use	Yes _ __ No_ X __	
F. Other Justification		

Notes to Proforma

A. Business Relevance. That which identifies market place relevance in terms of what problem is being solved and or need being addressed.

A.1 Market Requirement. When submitting a NP, the proposer shall identify the nature of the Market Requirement, assessing the extent to which it is essential, desirable or merely supportive of some other project.

A.2 Technical Regulation. If a Regulatory requirement is deemed to exist - e.g. for an area of public concern e.g. Information Security, Data protection, potentially leading to regulatory/public interest action based on the use of this voluntary international standard - the proposer shall identify this here.

B. Related Work. Aspects of the relationship of this NP to other areas of standardisation work shall be identified in this section.

B.1 Competition/Maintenance. If this NP is concerned with completing or maintaining existing standards, those concerned shall be identified here.

B.2 External Commitment. Groups, bodies, or for external to JTC 1 to which a commitment has been made by JTC for Co-operation and or collaboration on this NP shall be identified here.

B.3 External Std/Specification. If other activities creating standards or specifications in this topic area are known to exist or be planned, and which might be available to JTC 1 as PAS, they shall be identified here.

C. Technical Status. The proposer shall indicate here an assessment of the extent to which the proposed standard is supported by current technology.

C.1 Mature Technology. Indicate here the extent to which the technology is reasonably stable and ripe for standardisation.

C.2 Prospective Technology. If the NP is anticipatory in nature based on expected or forecasted need, this shall be indicated here.

C.3 Models/Tools. If the NP relates to the creation of supportive reference models or tools, this shall be indicated here.

D. Conformity Assessment and Interoperability Any other aspects of background information justifying this NP shall be indicated here.

D.1 Indicate here if Conformity Assessment is relevant to your project. If so, indicate how it is addressed in your project plan.

D.2 Indicate here if Interoperability is relevant to your project. If so, indicate how it is addressed in your project plan

E. Adaptability to Culture, Language, Human Functioning and Context of Use

NOTE: The following criteria do not mandate any feature for adaptability to culture, language, human functioning or context of use. The following criteria require that if any ISO/IEC JTC 1Directives, 5th Edition Version 3.0, 99

features are provided for adapting to culture, language, human functioning or context of use by the new Work Item proposal, then the proposer is required to identify these features.

E.1 Cultural and Linguistic Adaptability. Indicate here if cultural and natural language adaptability is applicable to your project. If so, indicate how it is addressed in your project plan.

ISO/IEC TR 19764 (Guidelines, methodology, and reference criteria for cultural and linguistic adaptability in information technology products) now defines it in a simplified way: “ability for a product, while keeping its portability and interoperability properties, to:

- be internationalized, that is, be adapted to the special characteristics of natural languages and the commonly accepted rules for their se, or of cultures in a given geographical region;

- take into account the usual needs of any category of users, with the exception of specific needs related to physical constraints”

Examples of characteristics of natural languages are: national characters and associated elements (such as hyphens, dashes, and punctuation marks), writing systems, correct transformation of characters, dates and measures, sorting and searching rules, coding of national entities (such as country and currency codes), presentation of telephone numbers and keyboard layouts. Related terms are localization, jurisdiction and multilingualism.

E.2 Adaptability to Human Functioning and Context of Use. Indicate here whether the proposed standard takes into account diverse human functioning and diverse contexts of use. If so, indicate how it is addressed in your project plan.

NOTE:

1. Human functioning is defined by the World Health Organization at http://www3.who.int/icf/beginners/bg.pdf as:

<<In ICF (International Classification of Functioning, Disability and Health), the term functioning refers to all body functions, activities and participation.>>

2. Content of use is defined in ISO 9241-11:1998 (Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability) as:

<<Users, tasks, equipment (hardware, software and materials), and the physical and societal environments in which a product is used.>>

3. Guidance for Standard Developers to address the needs of older persons and persons with disabilities).

F. Other Justification Any other aspects of background information justifying this NP shall be indicated here.

ITU
International Telecommunication Union
Telecommunication Standardization Sector

ISO
International Organization for Standardization

IEC
International Electrotechnical Commission

Patent Statement and Licensing Declaration

for a common text or twin text ITU-T Recommendation | ISO/IEC International Standard

This form is only to be used for such common texts or twin texts
This declaration does not represent an implied license grant

Please return to **each** of the three organizations:

Director
Telecommunication Standardization Bureau
International Telecommunication Union
Place des Nations
CH-1211 Geneva 20,
Switzerland
Fax: +41 22 730 5853

Secretary-General
International Organization for Standardization
1 rue de Varembe
CH-1211 Geneva 20
Switzerland
Fax: +41 22 733 3430

General Secretary
International Electrotechnical Commission
3 rue de Varembe
CH-1211 Geneva 20
Switzerland
Fax: +41 22 919 0301

(version: 24 November 2003)

Patent Holder/Organization:

Legal Name: CHINA IWNCOMM CO., LTD.

Contact for license application:

Name & Department: Dou Zhenyang, Management Department

Address: A201, QinFeng Ge, Xi'an Software Park, No. 68 KeJi 2nd Road, Xi'an Hi-Tech Industrial Development Zone, Xi'an, Shaanxi, P.R.China 710075

Tel.: +86 29 87607834

Fax: +86 29 87607829

E-mail: std@iwncomm.com

ITU-T Recommandation | ISO/IEC International Standard:

Number

Title: Information technology-Telecommunications and information exchange between system-Local and Mestropolitan area networks-Specific requirements-Part XX: Alternative security mechanism for use with ISO/IEC 8802-11

Licensing declaration

The Patent Holder believes that it holds granted patents and/or pending applications, the use of which would be required to implement the above ITU-T Recommendation | ISO/IEC International Standard and hereby declares, in accordance with the Statement on ITU-T Patent Policy (see ITU-T web site) and the ISO/IEC Patent Policy (JTC 1 Directives), that (check one box only).

☐ 1. The Patent Holder will grant a royalty-free license to an unrestricted number of applicants on a worldwide, non-discriminatory basis to use the patented material necessary in order to manufacture, use, and/or sell implementations of the above ITU-T Recommendation | ISO/IEC International Standard. Mark here __ if the Patent Holder's willingness to license is conditioned on reciprocity for the above ITU-T Recommendation | ISO/IEC International Standard.*

☑ 2. The Patent Holder will grant a license to an unrestricted number of applicants on a worldwide, non-discriminatory basis and on reasonable terms and conditions to use the patented material necessary in order to manufacture, use, and/or sell implementations of the above ITU-T Recommendation | ISO/IEC International Standard. Mark here __ if the Patent Holder's willingness to license is conditioned on reciprocity for the above ITU-T Recommendation | ISO/IEC International Standard.*

Negotiations of licenses are left to the parties concerned and are performed outside the ITU-T | ISO/IEC.

☐ 3. The Patent Holder is unwilling to grant licenses in accordance with provisions of either 1 or 2 above. In this case, the following information must be provided as part of this declaration:

- patent registration/application number;
- an indication of which portions of the ITU-T Recommendation | ISO/IEC International Standard

are affected.

- a description of the patent claims covering the ITU-T Recommendation | ISO/IEC International Standard;

* "Reciprocity" means with respect to other parties that have a patent or patent claim required in the use or implementation of the relevant ITU-T Recommendation(s) | ISO/IEC International Standard(s), the Patent Holder shall only be required to license to such parties if they are willing to license their patents or patent claims under options 1 or 2 of the Patent Statement and Licensing Declaration.

Signature

Organization	CHINA IWNCOMM CO., LTD.
Name of authorized person	Cao. Jun
Title of authorized person	General Manager
Signature	
Place, Date	Xi'an, 2009-7-20

ISO/IEC JTC 1/SC 6 N

Date: 2009-06-10

ISO/IEC NP XXXXX

ISO/IEC JTC 1/SC 6/WG

Secretariat: SAC

Information technology —Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part XX: Alternative security mechanism for use with ISO/IEC 8802-11

Élément introductif — Élément central — Partie XX: Élément complémentaire

Document type: International Standard
Document subtype:
Document stage: (20) Preparatory
Document language: E

C:\【最终版】附件 3 新工作项目提案文本（WAPI）-ISO_IEC XXXXX-Alternative security mechanism-英文.doc STD Version 2.1c2

Contents

Page

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee 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 drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC XXXXX was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*.

ISO/IEC 8802 consists of the following parts, under the general title *Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements*:

Part 1: Overview of Local Area Network Standards

Part 2: Logical link control

Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications

Part 5: Token ring access method and physical layer specifications

Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications

Part XX: Alternative security mechanism for use with ISO/IEC 8802-11

Introduction

This specification is a part of ISO/IEC 8802, and specifies an alternative security mechanism for use with ISO/IEC 8802-11. In this document, “ISO/IEC 8802-11” indicates the following four documents: ISO/IEC 8802-11:2005, *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,* and its AMENDMENT 4 (2006): *Further Higher Data Rate Extension in the 2.4 GHz Band*, AMENDMENT 5 (2006): *Spectrum and Transmit Power Management Extensions in the 5 GHz band in Europe*, and AMENDMENT 6 (2006): *Medium Access Control (MAC) Security Enhancements*.

Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements — Part XX: Alternative security mechanism for use with ISO/IEC 8802-11

1 Scope

ISO/IEC 8802 is a family of standards for for local and metropolitan area networks. Specifically, in this part of ISO/IEC 8802, a security mechanism, WLAN Authentication and Privacy Infrastructure (WAPI), is defined.

The purpose of this part of ISO/IEC 8802 is to provide an alternative security mechanism for use with ISO/IEC 8802-11, for secure wireless connectivity to automatic machinery, equipment or stations that require rapid deployment, which may be portable or hand-held, or which may be mounted on moving vehicles within a local area. For the coexistence of different security mechanisms, this part of ISO/IEC 8802 mainly provides an enhanced security mechanism (WAPI), which is different from those of ISO/IEC 8802-11:2005, Amd 4:2006, Amd 5:2006 and Amd 6:2006.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 8802-11:2005 *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*.

ISO/IEC 8802-11:2005/Amd 4:2006, *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, AMENDMENT 4: Further Higher Data Rate Extension in the 2.4 GHz Band*.

ISO/IEC 8802-11:2005/Amd 5:2006, *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, AMENDMENT 5: Spectrum and Transmit Power Management Extensions in the 5 GHz band in Europe*.

ISO/IEC 8802-11:2005/Amd 6:2006, *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, AMENDMENT 6: Medium Access Control (MAC) Security Enhancements*.

ISO/IEC 10116:2006, *Information technology — Security techniques — Modes of operation for an n-bit block cipher*

ANSI X9.62, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*

China State Cryptography Administration*, ECC and Parameters of ECDSA and ECDH Cryptography in WLAN Products,* 2006.

China State Cryptography Administration, *SMS4 Cryptography in WLAN Products*, 2006.

FIPS 180-2, *Secure Hash Standard (SHS)*, August 2002.

IEEE Std 802-1990, *IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture*.

IEEE Std 802.1X™-2004, *IEEE Standard for Local and Metropolitan Area Networks: Port-Based Network Access Control.*

ITU-T Recommendation X.509, ISO/IEC 9594, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks*.

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 8802-11 and the following apply.

3.1
authentication and key management (AKM) suite
set of one or more algorithms designed to provide authentication and key management, either individually or in combination with higher layer authentication and key management algorithms outside the scope of this part of ISO/IEC 8802

3.2
authenticator entity
AE
entity that facilitates an authentication action of the authentication supplicant entity (ASUE) before the ASUE accesses the network

NOTE The AE resides in a station (STA) or an access point (AP).

3.3
authentication service entity
ASE
entity that provides mutual identity authentication service for the authenticator entity (AE) and the authentication supplicant entity (ASUE) according to the credentials provided by the AE and ASUE

NOTE The ASE resides in any authentication service unit (ASU).

3.4
authentication service unit
ASU
important part of the wireless local area network (WLAN) authentication infrastructure (WAI) authentication framework, based on a public-key cryptography, which manages certificates and authenticates the users' identity

3.5
authentication supplicant entity
ASUE
entity that requires an identity authentication through the authentication service unit (ASU)

NOTE The ASUE resides in any station (STA).

3.6
base key
BK
highest order key used to generate the unicast session key (USK), either derived from a wireless local area network (WLAN) authentication infrastructure (WAI) Certificate Authentication procedure or directly derived from the preshared key (PSK)

3.7
base key security association
BKSA
result of the wireless local area network (WLAN) authentication infrastructure (WAI) Certificate Authentication or the result directly derived from the preshared key (PSK)

3.8
cryptographic system
cryptosystem
collection of transformations from plain text into ciphertext and vice versa, the particular transformation(s) to be used being selected by keys

NOTE The transformations are normally defined by a mathematical algorithm.

3.9
key encryption key
KEK
key used to encrypt the key data field in the key management protocol

3.10
lifetime
permitted period from start to timeout

3.11
message authentication key
MAK
key used to authenticate the data source and check integrity in the key management protocol

3.12
multicast session key
MSK
random value used to protect multicast medium access control (MAC) protocol data units (MPDUs) sent by the node

NOTE The MSK is derived from the multicast master key, including multicast encryption key (MEK) and multicast integrity check key (MCK).

3.13
multicast session key security association
MSKSA
result of the Multicast Key Announcement

3.14
notification master key
NMK
auxiliary key used to generate the multicast encryption key (MEK) and multicast integrity check key (MCK)

NOTE NMK is used as a multicast master key in the Multicast Key Announcement and a station-to-station (STA-to-STA) master key in the STA-to-STA STAKey Announcement, respectively

3.15
private key
<public-key cryptosystem> that key of a user's key pair which is known only by that user

3.16
public-key
<public-key cryptosystem> that key of a user's key pair which is publicly known

3.17
rekeying
key update process

3.18
STAKey
symmetric key used to protect direct communications between two stations (STAs) in a basic service set (BSS), including the STA-to-STA encryption key and STA-to-STA integrity check key

NOTE The STAKey is derived from a station-to-station (STA-to-STA) master key.

3.19
STAKey security association
STAKeySA
result of the station-to-station (STA-to-STA) STAKey Announcement in a basic service set (BSS), including a STAKey

3.20
unicast session key
USK
random value that is derived from the base key (BK) using a pseudo-random function, and composed of four parts: unicast encryption key (UEK), unicast integrity check key (UCK), message authentication key (MAK) and key encryption key (KEK)

3.21
unicast session key security association
USKSA
result of the Unicast Key Negotiation

3.22
wireless local area network (WLAN) authentication and privacy infrastructure
WAPI
security mechanism that is stated to provide the identity authentication and data confidentiality for the WLAN, and comprising the WLAN authentication infrastructure (WAI) and the WLAN privacy infrastructure (WPI)

3.23
wireless local area network (WLAN) authentication and privacy infrastructure (WAPI) Controlled Port
type of WAPI Port that allows the exchange of the protocols (except WAI protocols) only if the current state of the port is authorized

3.24
wireless local area network (WLAN) authentication and privacy infrastructure (WAPI) key management
Key management that includes the Unicast Key Negotiation, the Multicast Key Announcement and the STAKey Announcement

3.25
wireless local area network (WLAN) authentication and privacy infrastructure (WAPI) Port
WAPI Port
port that determines when to allow data traffic across an ISO/IEC 8802-11 link, and comprising a WAPI Controlled Port and a WAPI Uncontrolled Port

3.26
wireless local area network (WLAN) authentication and privacy infrastructure (WAPI) security network
network with the WAPI security mechanism

NOTE This network is identified by the WAPI Parameter Set information element in management frames such as the Beacon frame.

3.27
wireless local area network (WLAN) authentication and privacy infrastructure (WAPI) Uncontrolled Port
type of WAPI Port that allows the uncontrolled exchange of WAI protocols between stations (STAs) within the WLAN, regardless of the authorization state

3.28
wireless local area network (WLAN) authentication infrastructure
WAI
security scheme that is used for the identity authentication and key management in a WLAN

3.29
wireless local area network (WLAN) authentication infrastructure (WAI) Certificate Authentication
certificate-based authentication and key negotiation method that establishes the base key security association (BKSA) in the WAI authentication mechanism

3.30
wireless local area network (WLAN) privacy infrastructure
WPI
security scheme that is used for data transport protection in a WLAN

4 Abbreviated terms

For the purposes of this part of ISO/IEC 8802, the following abbreviated terms apply.

ACK	acknowledgment
ADDID	address index
AE	authenticator entity
AKM	authentication and key management
AKMP	authentication and key management protocol
AP	access point
ASE	authentication service entity
ASN.1	abstract syntax notation one
ASU	authentication service unit

ASUE	authentication supplicant entity
BK	base key
BKID	base key identification
BKSA	base key security association
BSS	basic service set
BSSID	basic service set identification
CA	certification authority
CBC-MAC	cipher block chaining message authentication code
CCA	clear channel assessment
CRC	cyclic redundancy code
CFP	contention-free period
DA	destination address
DCF	distributed coordination function
DER	distinguished encoding rules
DS	distribution system
DSM	distribution system medium
DSS	distribution system service
ECC	ellipse curve cipher
ECDH	elliptic curve Diffie-Hellman
ECDSA	elliptic curve digital signature algorithm
ESS	extended service set
FCS	frame check sequence
FH	frequency hopping
GBW	guo biao wuxian (China’s national wireless standard)
IBSS	independent basic service set
IV	initialization vector
KEK	key encryption key
LAN	local area network
LLC	logical link control

LME	layer management entity
MAC	medium access control
MAK	message authentication key
MCK	multicast integrity check key
MEK	multicast encryption key
MIB	management information base
MIC	message integrity code
MLME	MAC sublayer management entity
MMPDU	MAC management protocol data unit
MPDU	MAC protocol data unit
MSDU	MAC service data unit
MSK	multicast session key
MSKID	multicast session key index
MSKSA	multicast session key security association
MTU	maximum transmission unit
NAV	network allocation vector
NMK	notification master key
OFB	output feedback
OID	object identifier
OSI	open system interconnection
OUI	organization unique identifier
PAE	port access entity
PCF	point coordination function
PDU	protocol data unit
PEM	privacy enhanced mail
PHY	physical layer
PSK	preshared key
RSNA	robust security network association
SAP	service access point

s	seconds
SME	station management entity
SMS4	SMS4 block cipher cryptosystem
SS	station service
STA	station
STAKeyID	STAKey index
STAKeySA	STAKey security association
TLV	type-length-value format
UCK	unicast integrity check key
UDP	user datagram protocol
UEK	unicast encryption key
USK	unicast session key
USKID	unicast session key index
USKSA	unicast session key security association
UTC	universal time coordinated
WAI	WLAN authentication infrastructure
WAPI	WLAN authentication and privacy infrastructure
WEP	wired equivalent privacy
WLAN	wireless local area network
WM	wireless medium
WPI	WLAN privacy infrastructure

5 General description

5.1 General description of the architecture

See paragraph 1 of 5.1 of ISO/IEC 8802-11:2005 and Amd 6:2006.

5.1.1 How wireless LAN systems are different

See paragraph 1 of 5.1.1 of ISO/IEC 8802-11:2005.

5.1.1.1 Destination address does not equal destination location

See 5.1.1.1 of ISO/IEC 8802-11:2005.

5.1.1.2 The media impact of the design

See 5.1.1.2 of ISO/IEC 8802-11:2005.

5.1.1.3 The impact of handling mobile stations

See 5.1.1.3 of ISO/IEC 8802-11:2005.

5.1.1.4 Interaction with other ISO/IEC 8802-11 layers

See paragraph 1 of 5.1.1.4 of ISO/IEC 8802-11:2005.

Within a WLAN authentication and privacy infrastructure (WAPI) security network, WLAN authentication infrastructure (WAI) is used to implement mutual authentication and provides a WAPI Port. WLAN privacy infrastructure (WPI) provides functions to protect data frames. All stations (STAs) have a corresponding WAI Port Access Entity (PAE) that handles these services.

5.1.1.5 Interaction with non-ISO/IEC 8802-11 protocols

Within a WAPI security network, WAI protocols utilize non-ISO/IEC 8802-11 protocols for their authentication and key management (AKM) services.

5.2 Components of the ISO/IEC 8802-11 architecture

See paragraphs 1 to 3 of 5.2 of ISO/IEC 8802-11:2005.

5.2.1 The independent BSS (IBSS) as an ad hoc network

See 5.2.1 of ISO/IEC 8802-11:2005.

5.2.2 Distribution system (DS) concepts

See paragraphs 1 to 8 of 5.2.2 of ISO/IEC 8802-11:2005.

5.2.2.1 Extended service set (ESS): The large coverage network

See 5.2.2.1 of ISO/IEC 8802-11:2005.

5.2.2.2 WAPI

The WAPI mechanism defines a number of security features which differ from the robust security network association (RSNA). These features include the following:

a) mutual authentication mechanisms for STAs;

b) key management algorithms;

c) cryptographic key establishment;

d) enhanced data encapsulation mechanism.

The WAPI mechanism relies on two components external to the ISO/IEC 8802-11 architecture:

a) The first component is a WAI port access entity (PAE). PAEs are present on all STAs and control the forwarding of data to and from the medium access control (MAC). A WAI PAE that performs the

authenticator entity (AE) role, such as an AP in an authentication exchange, is known as an AE PAE; A WAI PAE that performs the authentication supplicant entity (ASUE) role, such as a non-AP STA in an authentication exchange, is known as an ASUE PAE. In an IBSS, each STA has both an AE PAE and an ASUE PAE associated with it.

b) The second component is the authentication service unit (ASU). The ASU performs the necessary mutual authenticaiton function to check credentials of not only the ASUE such as a non-AP STA on behalf of the AE, but also the AE such as an AP on behalf of the ASUE. In certain applications, the ASU may be integrated into the same physical device as the AP.

5.2.3 Area concepts

See 5.2.3 of ISO/IEC 8802-11:2005.

5.2.4 Integration with wired LANs

See 5.2.4 of ISO/IEC 8802-11:2005.

5.3 Logical service interfaces

See 5.3 of ISO/IEC 8802-11:2005, Amd 5:2006 and Amd 6:2006.

5.4 Overview of the services

See paragraphs 1 to 7 of 5.4 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.4.1 Distribution of messages within a DS

See 5.4.1 of ISO/IEC 8802-11:2005.

5.4.2 Services that support the distribution service

See paragraphs 1 and 2 of 5.4.2 of ISO/IEC 8802-11:2005.

5.4.2.1 Mobility types

See 5.4.2.1 of ISO/IEC 8802-11:2005.

5.4.2.2 Association

See paragraphs 1 and 2 of 5.4.2.2 of ISO/IEC 8802-11:2005.

Within a WAPI security network, this is different. In WAPI, the WAPI Port determines when to allow data traffic across an ISO/IEC 8802-11 link. A single WAPI Port maps to one association, and each association maps to a WAPI Port. A WAPI Port consists of a WAPI Controlled Port and a WAPI Uncontrolled Port. The WAPI Controlled Port is blocked from passing general data traffic between two STAs until a WAI authentication procedure completes successfully over the WAPI Uncontrolled Port. Once the AKM completes successfully, data protection is enabled to prevent unauthorized access, and the WAPI Controlled Port unblocks to allow protected data traffic. ASUEs and AEs exchange protocol information via the WAPI Uncontrolled Port. It is expected that most other protocol exchanges will make use of the WAPI Controlled Ports. However, a given protocol may need to bypass the authorization function and make use of the WAPI Uncontrolled Port.

See paragraphs 4 and 5 of 5.4.2.2 of ISO/IEC 8802-11:2005.

5.4.2.3 Reassociation

See paragraphs 1 and 2 of 5.4.2.3 of ISO/IEC 8802-11:2005.

If no facilities are provided to move a WAPI process during reassociation, then the old WAPI process will be deleted, and a new WAPI process will need to be constructed.

5.4.2.4 Disassociation

See 5.4.2.4 of ISO/IEC 8802-11:2005.

5.4.3 Access control and confidentiality services

See paragraphs 1 and 2 of 5.4.3 of ISO/IEC 8802-11:2005 and Amd 6:2006.

In a wireless LAN that does not support WAPI, two services, authentication and confidentiality, are defined. ISO/IEC 8802-11 authentication is used instead of the wired media physical connection. WEP encryption was defined to provide the confidentiality aspects of closed wired media.

Within a WAPI security network, the WAPI Controlled Port along with WAI protocols is used to provide access control. The WAPI station management entity (SME) provides key management via an exchange of WAI protocol frames. Confidentiality and data integrity are provided by WPI.

5.4.3.1 Authentication

See paragraphs 1 to 3 of 5.4.3.1 of ISO/IEC 8802-11:2005.

Within a WAPI security network, Open System（see ISO/IEC 8802-11:2005）authentication is used to test the connectivity at the link level, which may be established before association. It disallows the use of Shared Key（see ISO/IEC 8802-11:2005）authentication.

WAPI supports the authentication based on WAI which is used to implement mutual identity authentication between two STAs and is established after association. In a BSS, when the identity authentication is successful, the STA will be authorized to securely access to the AP; otherwise, both the AP and STA refuse to communicate with each other. In an IBSS, when the identity authentication is successful, the initiator STA will be authorized to securely access the peer STA; otherwise, STAs refuse to communicate with each other.

WAI supports the identity authentication based on certificates (WAI Certificate Authentication), or preshared keys (PSKs). The WAI Certificate Authentication utilizes public-key certificates to authenticate STAs and APs.

In WAPI, the WAI ASUE and AE exchange protocol information via the WAPI Uncontrolled Port. The WAPI Controlled Port is blocked from passing general data traffic between two STAs until a WAI authentication procedure completes successfully over the WAPI Uncontrolled Port. See Clause 8 for more details.

In a WAPI BSS or ESS, Open System authentication shall be required for testing the connectivity at the link level before association. In a WAPI IBSS, Open System authentication should optionally be performed before association.

See paragraphs 7 to 8 of 5.4.3.1 of ISO/IEC 8802-11:2005.

5.4.3.2 Deauthentication

See paragraphs 1 and 2 of 5.4.3.2 of ISO/IEC 8802-11:2005.

Within a WAPI security network, the deauthentication service is invoked when an existing Open System authentication is to be terminated. Deauthentication is an SS.

In a WAPI ESS, Open System authentication shall be required. In a WAPI ESS, deauthentication results in termination of any association for the deauthenticated station. It also results in the WAPI Controlled Port for that STA being disabled and deletes the unicast session key security association (USKSA). The deauthentication notification is provided to WAI via the MAC layer.

In WAPI, deauthentication also destroys any related USKSA, multicast session key security association (MSKSA), and STAKey security associations (STAKeySAs) that exist in the STA and closes the associated WAI Controlled Port. If base key (BK) caching is disabled, deauthentication also destroys the base key security association (BKSA) from which the deleted USKSA was derived.

In a WAPI IBSS, Open System authentication is optional, but a STA is required to recognize Deauthentication frames. Deauthentication results in the WAPI Controlled Port for that STA being disabled and deletes the USKSA.

5.4.3.3 Confidentiality

See paragraphs 1 to 3 of 5.4.3.3 of ISO/IEC 8802-11:2005.

This part of ISO/IEC 8802 specifies a confidentiality algorithm, WPI, that is designed to satisfy the goal of wired LAN “equivalent” confidentiality. The algorithm is not designed for ultimate security but rather to be “at least as secure as a wire”. See Clause 8 for more details.

This part of ISO/IEC 8802 uses the WPI mechanism (see Clause 8) to perform the actual encryption of messages. Management information base (MIB) functions are provided to support WPI.

Note that confidentiality can only be invoked for data frames. All stations initially start “in the clear” in order to set up the authentication and confidentiality services.

See the last paragraph of 5.4.3.3 of ISO/IEC 8802-11:2005.

5.4.3.4 Key management

The enhanced confidentiality, data authentication and replay protection mechanisms require establishing fresh cryptographic keys. The procedures defined in this part of ISO/IEC 8802 provide keys by means of processes called the Unicast Key Negotiation, Multicast Key Announcement and STAKey Announcement.

5.4.3.5 Data origin authenticity

The data origin authenticity mechanism defines a means by which a STA that receives a data frame can determine which STA transmitted the MAC protocol data unit (MPDU). This feature is required in WAPI to prevent one STA from masquerading as a different STA. This mechanism is provided for STAs that use WPI-SMS4 (see Clause 8) to perform the actual encryption of messages. MIB functions are provided to support WPI.

See paragraph 2 of 5.4.3.5 of ISO/IEC 8802-11:2005.

NOTE SMS4 is instantiated in this standard. The ciphersuite can be instantiated with any other 128-bit block cipher.

5.4.3.6 Replay detection

The replay detection mechanism defines a means by which a STA that receives a data frame from another STA can detect whether the data frame is an unauthorized retransmission. This mechanism is provided for STAs that use WPI-SMS4 (see Clause 8) to perform the actual encryption of messages. MIB functions are provided to support WPI.

5.4.4 Spectrum management services

See 5.4.4 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.5 Relationships between services

See 5.5 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.6 Differences between ESS and IBSS LANs

See 5.6 of ISO/IEC 8802-11:2005 and Amd 6:2006.

5.7 Message information contents that support the services

See paragraph 1 of 5.7 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.7.1 Data

See 5.7.1 of ISO/IEC 8802-11:2005.

5.7.2 Association

See 5.7.2 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.7.3 Reassociation

See 5.7.3 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.7.4 Disassociation

See 5.7.4 of ISO/IEC 8802-11:2005.

5.7.5 Confidentiality

See 5.7.5 of ISO/IEC 8802-11:2005.

5.7.6 Authentication

See paragraphs 1 and 2 of 5.7.6 of ISO/IEC 8802-11:2005.

This sepcification defines a subtype of authentication service: Open System. The subtype invoked is indicated in the body of authentication management frames. Thus authentication frames are selfidentifying with respect to authentication algorithm. All management frames of subtype Authentication shall be unicast frames as authentication is performed between pairs of stations (i.e., multicast authentication is not allowed). Management frames of subtype Deauthentication are advisory, and may therefore be sent as group-addressed frames.

See 8.1.1 of 5.7.6 of ISO/IEC 8802-11:2005.

5.7.7 Deauthentication

See 5.7.7 of ISO/IEC 8802-11:2005.

5.7.8 Spectrum management

See 5.7.8 of ISO/IEC 8802-11:2005 and Amd 5:2006.

5.8 Reference model

See paragraph 1 of 5.8 of ISO/IEC 8802-11:2005.

There is an interface between the WAPI ASUE/AE shown in Figure 1.

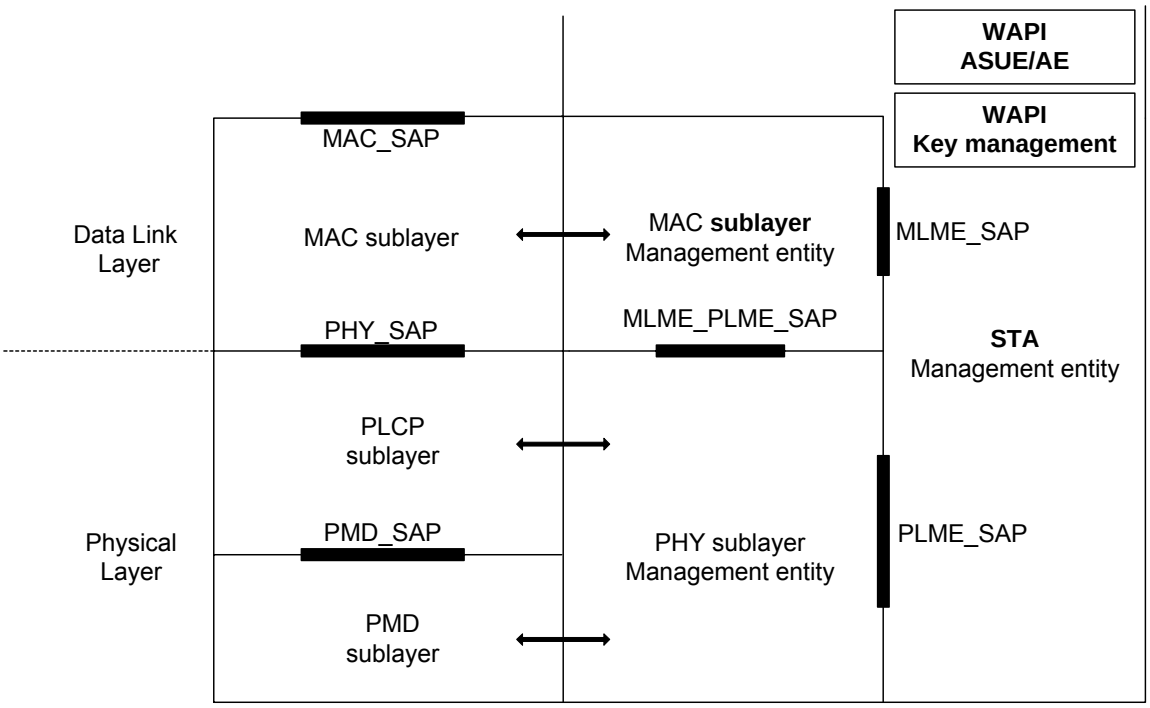


Figure 1 — Portion of the ISO/IEC basic reference model covered in this part of ISO/IEC 8802

5.9 Establishing the security association

5.9.1 Infrastructure mode

A STA discovers the AP's security policy through passively monitoring Beacon frames or through active probing (shown in Figure 2). If the WAI certificate Authentication and Key Management Protocol (AKMP) is used, the AP sends a WAI Authentication Activation message to start the WAI Certificate Authentication. After this process, the Unicast Key Negotiation and Multicast Key/STAKey Announcement are executed between the AP and STA. If the PSK mechanism is used, the Unicast Key Negotiation and Multicast Key/STAKey Announcement are directly executed between the AP and STA.

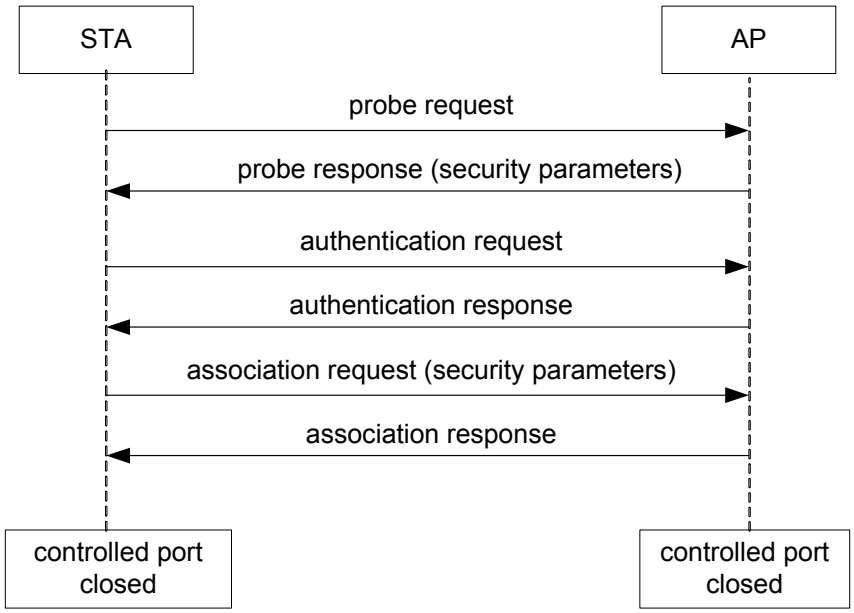


Figure 2 — The establishment of security association in infrastructure mode

5.9.2 IBSS mode

5.9.2.1 The initiation of authentication based on the PSK negotiation procedure in an IBSS

In an IBSS, 5-Way handshakes are carried out two times (the first three messages are used in the Unicast Key Negotiation and the last two are used in the Multicast Key Announcement) between two STAs. Two independent unicast session keys (USKs) will be negotiated and their own multicast session key (MSK) will be announced to each other. During the practical communication, for a pair of STAs, the unicast data between the two STAs are encrypted or decrypted using the unicast encryption key (UEK) and the unicast integrity check key (UCK) which are generated by the process initiated by the STA (as AE) with the higher MAC address, whereas the keys (UEK and UCK) generated by the process initiated by the STA (as AE) with the lower MAC address will not be used. Each STA encrypts its broadcast/multicast data using the MSK announced by itself before sending them and the other STAs decrypt these data using the same key after receiving them.

The reasons or conditions for the initiation of the Unicast Key Negotiation based on the PSK are listed below.

a) When a STA, e.g. named STA1, receives a Beacon frame or a Probe Response frame from another STA, e.g. named STA2, which has not completed a Unicast Key Negotiation with STA1, a Unicast Key Negotiation will be initiated.

b) When STA1 receives the first protocol frame of the Unicast Key Negotiation from STA2 which has not completed a Unicast Key Negotiation with STA1, a Unicast Key Negotiation will be initiated.

c) When STA1 receives a WPI encapsulation broadcast/multicast data from STA2, if STA1 does not have the corresponding decryption key, STA1 shall notify the station management entity (SME) via an MLME-PROTECTEDFRAMEDROPPED.indication primitive. If the SME receives this primitive and has not completed a Unicast Key Negotiation with STA2 specified in the primitive, a Unicast Key Negotiation will be initiated.

5.9.2.2 The initiation of authentication based on the certificate in an IBSS

When the authentication method based on a certificate is used between two STAs, each STA initiates its own WAI Certificate Authentication, Unicast Key Negotiation and Multicast Key Negotiation, respectively. After

these processes, they establish two independent BKs as well as two independent USKs and announce its own MSK to each other. During the practical communication, for a pair of STAs, the unicast data between the two STAs are encrypted or decrypted using the unicast encryption key (UEK) and the unicast integrity check key (UCK) which are generated by the process initiated by the STA (as AE) with the higher MAC address, whereas the keys (UEK and UCK) generated by the process initiated by the STA (as AE) with the lower MAC address will not be used. Each STA encrypts its broadcast/multicast data using the MSK announced by itself before sending them, and the other STAs decrypt these data using the same key after receiving them.

The reasons or conditions for the initiation of the authentication process based on certificate are listed below.

a) When STA1 receives a Beacon frame or a Probe Response frame from STA2 which has not completed a WAI Certificate Authentication with STA1, a WAI Certificate Authentication will be initiated.

b) When STA1 receives the first protocol information of the WAI Certificate Authentication from STA2 which has not completed a WAI Certificate Authentication procedure with STA1, a WAI Certificate Authentication will be initiated.

c) When STA1 receives a WPI encapsulation broadcast/multicast data from STA2, if STA1 has not the corresponding decryption key, STA1 shall notify the SME via an MLME-PROTECTEDFRAMEDROPPED.indication primitive; if the SME receives this primitive and has not completed a WAI Certificate Authentication with STA2 described in the primitive, a WAI Certificate Authentication will be initiated.

If the WAPI security parameters of STA2 are unknown to STA1, STA1 should send a Probe Request frame to STA2 before the initiation of authentication. Then the security mechanism can be obtained from the Probe Response frame of STA2.

6 MAC service definition

6.1 Overview of MAC services

6.1.1 Asynchronous data service

See 6.1.1 of ISO/IEC 8802-11:2005.

6.1.2 Security services

Security services in this part of ISO/IEC 8802 are provided by the WAI authentication services and the WPI confidential mechanism. The scope of the security services provided is limited to station-to-station data exchange. The confidentiality service offered is the encryption of the MSDU. For the purposes of this part of ISO/IEC 8802, WPI is viewed as a logical service located within the MAC sublayer as shown in the reference model, Figure 1. Actual implementations of the WPI service are transparent to the LLC and other layers above the MAC sublayer.

The security services provided by this part of ISO/IEC 8802 are as follows:

a) confidentiality;

b) authentication; and

c) access control in conjunction with layer management.

During the authentication exchange, both parties exchange authentication information as described in Clause 8.

The MAC sublayer security services provided by WPI rely on information from nonlayer-2 management or system entities. Management entities communicate information to WPI through a set of MIB attributes.

6.1.3 MSDU ordering

See 6.1.3 of ISO/IEC 8802-11:2005.

6.1.4 MAC Data Service Infrastructure

The MAC data plane architecture（i.e. processes that involve transport of all or part of an MSDU procedure）is shown in Figure 3.

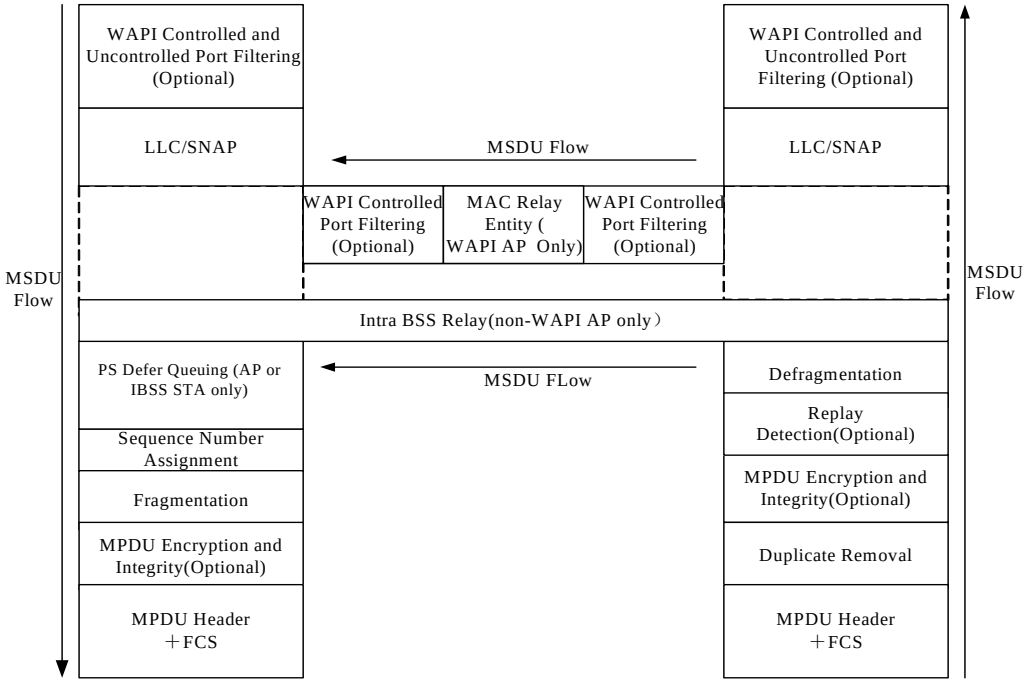


Figure 3 — MAC data plane architecture

During transmission, an MSDU goes through some or all of the following processes:

a) frame delivery deferral during power save mode,

b) sequence number assignment,

c) fragmentation,

d) encryption,

e) integrity protection, and

f) frame formation.

WAPI may block the MSDU at the WAPI Controlled Port.

During reception, a received data frame goes through the processes of

a) MPDU header + frame check sequence (FCS) validation,

b) duplicate removal,

c) decryption,

d) defragmentation,

e) integrity checking, and

f) replay detection.

The WAPI Controlled/ Uncontrolled Ports discard the MSDU if the WAPI Controlled Port is not enabled or if the MSDU does not represent a WAPI frame. WPI-SMS4 MPDU frame order enforcement occurs prior to MSDU defragmentation; therefore, defragmentation will fail if MPDUs arrive out of order.

6.2 Detailed service specification

See 6.2 of ISO/IEC 8802-11:2005.

7 Frame formats

See paragraph 1 of 7 of ISO/IEC 8802-11:2005.

7.1 MAC frame formats

See paragraph 1 of 7.1 of ISO/IEC 8802-11:2005.

7.1.1 Conventions

See 7.1.1 of ISO/IEC 8802-11:2005.

7.1.2 General frame format

See 7.1.2 of ISO/IEC 8802-11:2005.

7.1.3 Frame fields

7.1.3.1 Frame Control field

See 7.1.3.1 of ISO/IEC 8802-11:2005 and Amd 6:2006.

7.1.3.1.1 Protocol Version field

See 7.1.3.1.1 of ISO/IEC 8802-11:2005.

7.1.3.1.2 Type and Subtype fields

See 7.1.3.1.2 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.1.3.1.3 To DS field

See 7.1.3.1.3 of ISO/IEC 8802-11:2005.

7.1.3.1.4 From DS field

See 7.1.3.1.4 of ISO/IEC 8802-11:2005.

7.1.3.1.5 More Fragments field

See 7.1.3.1.5 of ISO/IEC 8802-11:2005.

7.1.3.1.6 Retry field

See 7.1.3.1.6 of ISO/IEC 8802-11:2005.

7.1.3.1.7 Power Management field

See 7.1.3.1.7 of ISO/IEC 8802-11:2005.

7.1.3.1.8 More Data field

See 7.1.3.1.8 of ISO/IEC 8802-11:2005.

7.1.3.1.9 Protected Frame field

The Protected Frame field shall be 1 bit in length. The Protected Frame field is set to 1 if the Frame Body field contains information that has been processed. The Protected Frame field shall only be set to 1 within data frames. The Protected Frame field shall be set to 0 in all other frames. In a data frame, the Frame Body field is protected utilizing the cryptographic encapsulation algorithm and expanded as defined in Clause 8.

7.1.3.1.10 Order field

See 7.1.3.1.10 of ISO/IEC 8802-11:2005.

7.1.3.2 Duration/ID field

See 7.1.3.2 of ISO/IEC 8802-11:2005.

7.1.3.3 Address fields

See 7.1.3.3 of ISO/IEC 8802-11:2005.

7.1.3.4 Sequence Control field

See 7.1.3.4 of ISO/IEC 8802-11:2005.

7.1.3.5 Frame Body field

See 7.1.3.5 of ISO/IEC 8802-11:2005.

7.1.3.6 FCS field

See 7.1.3.6 of ISO/IEC 8802-11:2005.

7.2 Format of individual frame types

7.2.1 Control frames

See 7.2.1 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.2.2 Data frames

See 7.2.2 of ISO/IEC 8802-11:2005 and Amd 6:2006.

7.2.3 Management frames

See paragraphs 1 to 9 of 7.2.3 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.2.3.1 Beacon frame format

The frame body of a management frame of subtype Beacon contains the information shown in Table 1.

Table 1 — Beacon frame body

Order	Information	Notes
22	WAPI Parameter Set	The WAPI Parameter Set information element is only present within Beacon frames generated by STAs when the WAPI mechanism is enabled.
Other	See Table 5 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006	

7.2.3.2 IBSS Announcement Traffic Indication Message (ATIM) frame format

See 7.2.3.2 of ISO/IEC 8802-11:2005.

7.2.3.3 Disassociation frame format

See 7.2.3.3 of ISO/IEC 8802-11:2005.

7.2.3.4 Association Request frame format

The frame body of a management frame of subtype Association Request contains the information shown in Table 2.

Table 2 — Association Request frame body

Order	Information	Notes
9	WAPI Parameter Set	The WAPI Parameter Set information element is only present within Association request frames generated by STAs when the WAPI mechanism is enabled.
Other	See Table 7 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006	

7.2.3.5 Association Response frame format

The frame body of a management frame of subtype Association Response contains the information shown in Table 3.

Table 3 — Association Response frame body

Order	Information	Notes
9	WAPI Parameter Set	The WAPI Parameter Set information element is only present within Association request frames generated by STAs when the WAPI mechanism is enabled.
Other	See Table 7 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006	

7.2.3.6 Reassociation Request frame format

The frame body of a management frame of subtype Reassociation Request contains the information shown in Table 4.

Table 4 — Reassociation Request frame body

Order	Information	Notes
10	WAPI Parameter Set	The WAPI Parameter Set information element is only present within Reassociation request frames generated by STAs when the WAPI mechanism is enabled.
Other	See Table 9 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006	

7.2.3.7 Reassociation Response frame format

See 7.2.3.7 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.2.3.8 Probe Request frame format

See 7.2.3.8 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.2.3.9 Probe Response frame format

The frame body of a management frame of subtype Probe Response contains the information shown in Table 5.

Table 5 — Probe Response frame body

Order	Information	Notes
22	WAPI Parameter Set	The WAPI Parameter Set information element is only present within Probe Response frames generated by STAs when the WAPI mechanism is enabled.

Other	See Table 12 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006	

7.2.3.10 Authentication frame format

The frame body of a management frame of subtype Authentication contains the information shown in

Table 6.

Table 6 — Probe Response frame body

Order	Information	Notes
1	Authentication algorithm number	
2	Authentication transaction sequence number	
3	Status code	The status code information is reserved and set to 0 in certain Authentication frames

Only Open System authentication shall be used within the WAPI mechanism for testing the connectivity at the link level.

7.2.3.11 Deauthentication

See 7.2.3.11 of ISO/IEC 8802-11:2005.

7.2.3.12 Action frame format

See 7.2.3.12 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3 Management frame body components

See paragraph 1 of 7.3 of ISO/IEC 8802-11:2005.

7.3.1 Fixed fields

7.3.1.1 Authentication Algorithm Number field

The Authentication Algorithm Number field indicates a single authentication algorithm. The length of the Authentication Algorithm Number field is 2 octets. The Authentication Algorithm Number field is illustrated in Figure 24 of ISO/IEC 8802-11:2005. The following values are defined for authentication algorithm number:

a) Authentication algorithm number = 0: Open System

b) All other values of authentication number are reserved.

7.3.1.2 Authentication Transaction Sequence Number field

See 7.3.1.2 of ISO/IEC 8802-11:2005.

7.3.1.3 Beacon Interval field

See 7.3.1.3 of ISO/IEC 8802-11:2005.

7.3.1.4 Capability Information field

See paragraphs 1 to 6 of 7.3.1.4 of ISO/IEC 8802-11:2005, Amd 4:2006 and Amd 5:2006.

APs within a BSS set the Privacy subfield to 1 within transmitted Beacon, Probe Response, Association Response and Reassociation Response management frames if data confidentiality is required for all data type frames exchanged within the BSS. If data confidentiality is not required, the Privacy subfield is set to 0.

In WAPI, non-AP STAs within an ESS set the Privacy subfield to 0 in transmitted Association or Reassociation management frames. APs ignore the Privacy subfield within received Association and Reassociation management frames.

STAs within an IBSS set the Privacy subfield to 1 in transmitted Beacon or Probe Response management frames if data confidentiality is required for all data type frames exchanged within the IBSS. If data confidentiality is not required, STAs in an IBSS set the Privacy subfield to 0 within these management frames.

STAs that include the WAPI Parameter Set information element in Beacon and Probe Response frames shall set the Privacy subfield to 1 in any frame that includes the WAPI Parameter Set information element.

See paragraphs 11 through the last paragraph of 7.3.1.4 of ISO/IEC 8802-11:2005 Amd 4:2006 and Amd 5:2006.

7.3.1.5 Current AP Address field

See 7.3.1.5 of ISO/IEC 8802-11:2005.

7.3.1.6 Listen Interval field

See 7.3.1.6 of ISO/IEC 8802-11:2005.

7.3.1.7 Reason Code field

See paragraph 1 of 7.3.1.7 of ISO/IEC 8802-11:2005.

The reason codes are defined in Table 7.

Table 7 — Reason codes

Reason code	Meaning
30	Unsupported WAPI Parameter Set information element version
31	Invalid WAPI Parameter Set information element capabilities
32	WAI Certificate Authentication failed
Other	See Table 17 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006

7.3.1.8 Association ID (AID) field

See 7.3.1.8 of ISO/IEC 8802-11:2005.

7.3.1.9 Status Code field

See paragraphs 1 and 2 of 7.3.1.9 of ISO/IEC 8802-11:2005. Status codes are shown in Table 8

Table 8 — Status codes

Status code	Meaning
49	Unsupported WAPI Parameter Set information element version
50	Invalid WAPI Parameter Set information element capabilities
Other	See Table 18 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006

7.3.1.10 Timestamp field

See 7.3.1.10 of ISO/IEC 8802-11:2005.

7.3.1.11 Action field

See 7.3.1.11 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2 Information elements

See paragraphs 1 and 2 of 7.3.2 of ISO/IEC 8802-11:2005. Information element IDs are shown in Table 9.

Table 9 — Element IDs

Information element	Element ID
WAPI Parameter Set	68
Other	See Table 19 of ISO/IEC 8802-11:2005 Amd 4:2006, Amd 5:2006 and Amd 6:2006

7.3.2.1 Service Set Identity (SSID) element

See 7.3.2.1 of ISO/IEC 8802-11:2005.

7.3.2.2 Supported Rates element

See 7.3.2.2 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.3.2.3 FH Parameter Set element

See 7.3.2.3 of ISO/IEC 8802-11:2005.

7.3.2.4 DS Parameter Set element

See 7.3.2.4 of ISO/IEC 8802-11:2005.

7.3.2.5 CF Parameter Set element

See 7.3.2.5 of ISO/IEC 8802-11:2005.

7.3.2.6 TIM

See 7.3.2.6 of ISO/IEC 8802-11:2005.

7.3.2.7 IBSS Parameter Set element

See 7.3.2.7 of ISO/IEC 8802-11:2005.

7.3.2.8 Challenge Text element

See 7.3.2.8 of ISO/IEC 8802-11:2005.

7.3.2.9 Country information element

See 7.3.2.9 of ISO/IEC 8802-11:2005.

7.3.2.10 Hopping Pattern Parameters information element

See 7.3.2.10 of ISO/IEC 8802-11:2005.

7.3.2.11 Hopping Pattern Table information element

See 7.3.2.11 of ISO/IEC 8802-11:2005.

7.3.2.12 Request information element

See 7.3.2.12 of ISO/IEC 8802-11:2005.

7.3.2.13 ERP Information element

See 7.3.2.13 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.3.2.14 Extended Supported Rates element

See 7.3.2.14 of ISO/IEC 8802-11:2005 and Amd 4:2006.

7.3.2.15 Power Constraint element

See 7.3.2.15 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.16 Power Capability element

See 7.3.2.16 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.17 TPC Request element

See 7.3.2.17 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.18 TPC Report element

See 7.3.2.18 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.19 Supported Channels element

See 7.3.2.19 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.20 Channel Switch Announcement element

See 7.3.2.20 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.21 Measurement Request element

See 7.3.2.21 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.22 Measurement Report element

See 7.3.2.22 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.23 Quiet element

See 7.3.2.23 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.24 IBSS DFS element

See 7.3.2.24 of ISO/IEC 8802-11:2005 and Amd 5:2006.

7.3.2.25 WAPI Parameter Set information element

The WAPI Parameter Set information element contains authentication and unicast cipher suite selectors, See Figure 4. All STAs implementing WAPI shall support this element. The size of the WAPI Parameter Set information element is limited by the size of an information element, which is 255 octets.

	Element Identifier ID	Length	Version	Authentication And Key Management (AKM) Suite Count	Authentication And Key Management (AKM) Suite	Unicast Cipher Suite Count	Unicast Cipher Suite	Multicast Cipher Suite	WAPI Capability Information	BKID Count	BKIDList
Octets:	1	1	2	2	4×m	2	4×n	4	2	2	16×s

Figure 4 — WAPI Parameter Set information element format

In Figure 4, m denotes AKM suite count, n is the unicast cipher suite count, and s is the BKID count. Other fields of Figure 4 are explained as follows.

a) The Element Identifier ID shall be 68 decimal.

b) The Length field specifies the number of octets in the information field [field(s) following the Element Identifier ID and Length fields] of the information element.

c) The Version field specifies the version number of the WAPI protocol. In this part of ISO/IEC 8802, this field should be set to 1, and other values are reserved.

d) The Authentication and Key Management (AKM) Suite Count field specifies the number of the authentication and key management mechanisms that the STA supports.

e) The Authentication and Key Management (AKM) Suite field contains the authentication and key management mechanisms that the STA supports.

f) The Unicast Cipher Suite Count field specifies the number of the unicast cipher algorithms that the STA supports.

g) The Unicast Cipher Suite field contains the unicast cipher algorithms that STA supports.

h) The Multicast Cipher Suite field contains the multicast cipher algorithms that the STA supports.

a) The WAPI Capability Information field indicates the requested or advertised capabilities. The WAPI Capability Information field shall be 2 octets in length, and the format of the WAPI Capability Information field shall be as illustrated in Figure 5 and described after the figure:

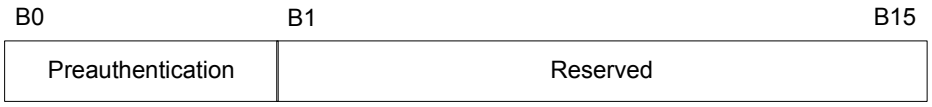


Figure 5 — WAPI Capability Information field format

— Bit 0: Preauthentication. An AP sets the preauthentication subfield of the WAPI Capability Information field to 1 when it supports preauthentication; otherwise, it will be set to 0. A non-AP STA sets the preauthentication subfield to 0.

— Others: Reserved.

a) The BKID Count and List fields are only present within Association or Reassociation Request frames sent to AP. The BKID Count field indicates the number of BKIDs in the BKID List field. The BKID List field contains 0 or more valid BKIDs that the STA believes to be valid for the destination AP. The BKIDs can refer to:

a) BKIDs generated by preauthentication between the current STA and the destination AP;

b) BKIDs generated by the WAI Certificate Authentication procedure;

c) BKIDs generated by the preshared key.

7.3.2.25.1 Authentication and Key Management suite

The Authentication and Key Management suite selector has the format shown in Figure 6.

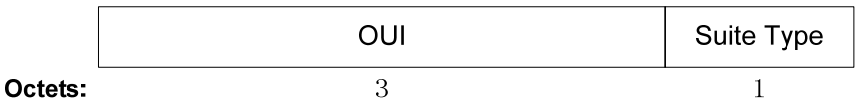


Figure 6 — Suite selector format

The suite selectors defined in this part of ISO/IEC 8802 are shown in Table 10.

Table 10 — Authentication and Key Management suite selectors

OUI 3 octets	Suite Type 1 octet	Meaning
00-14-72	0	Reserved
00-14-72	1	WAI Certificate Authentication and Key Management
00-14-72	2	WAI Preshared Key Authentication and Key Management
00-14-72	3-255	Reserved
Others	0-255	Reserved

7.3.2.25.2 Unicast cipher suite

The Unicast cipher suite selector format is shown in Figure 7.

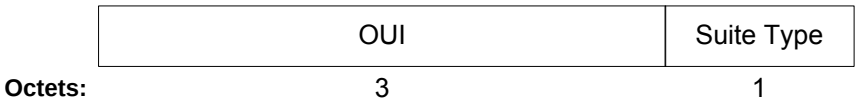


Figure 7 — Suite selector format

The suite selectors defined in this part of ISO/IEC 8802 are shown in Table 11.

Table 11 — Unicast cipher suite selectors

OUI 3 octets	Type 1 octet	Meaning
00-14-72	0	Reserved
00-14-72	1	WPI-SMS4
00-14-72	2~255	Reserved
others	0~255	Reserved

7.3.2.25.3 Multicast cipher suite

The suite selectors defined in this part of ISO/IEC 8802 are shown in Table 12.

Table 12 — Multicast cipher suite selectors

OUI	Type	Meaning

3 octets	1 octet	
00-14-72	0	Reserved
00-14-72	1	WPI-SMS4
00-14-72	2~255	Reserved
others	0~255	Reserved

7.4 Action frame format details

See 7.4 of ISO/IEC 8802-11:2005 and Amd 5:2006.

8 Security

The WAPI mechanism is adopted to implement access control and data confidentiality. The WAPI security comprises the following two services:

a) WAI authentication and key management, described in 8.1;

b) WPI data transport protection, described in 8.2.

8.1 WAI authentication and key management

The WAI authentication and key management is comprised of the four mechanisms described below.

a) Based on a certificate in a BSS:

   1) The non-AP STA identifies the Authentication and Key Management (AKM) suites supported by the AP through passively monitoring Beacon frames or through active probing (Probe Response frames).

   2) Open System authentication shall be invoked between the STA and AP.

   3) During (re)association, the STA determines the cipher suite based on the WAPI Parameter Set information element in the (Re)Association Request frame.

   4) The STA and AP invoke a WAI Certificate Authentication procedure to negotiate a BK.

   5) The STA and AP execute the Unicast Key Negotiation and Multicast Key/STAKey Announcement.

   6) The WPI algorithm uses the negotiated keys and cipher suites to protect the data.

b) Based on a PSK in a BSS:

   1) The non-AP STA identifies the Authentication and Key Management (AKM) suites supported by the AP through passively monitoring Beacon frame or through active probing (Probe Response frame).

   2) Open System authentication shall be invoked between the STA and AP.

   3) During (re)association, the STA determines the cipher suite based on the WAPI Parameter Set information element in the (Re) Association Request frame.

   4) The STA and an AP will execute the Unicast Key Negotiation and Multicast Key/STAKey Announcement after the BK is derived from a PSK.

5) The WPI algorithm uses the negotiated keys and cipher suites to protect the data.

c) Based on a certificate in an IBSS:

1) The initiator STA identifies the Authentication and Key Management (AKM) suites supported by the peer STA through passively monitoring Beacon frame or through active probing (Probe Response frame).

2) An optional Open System authentication may be invoked between the initiator and the peer STA.

3) The initiator and the peer STA invoke the WAI Certificate Authentication procedure to negotiate a BK.

4) The initiator and the peer STA execute the Unicast Key Negotiation and Multicast Key Announcement.

5) The WPI algorithm uses the negotiated keys and cipher suites to protect the data.

d) Based on a PSK in an IBSS :

1) The initiator STA identifies the Authentication and Key Management (AKM) suites supported by the peer STA through passively monitoring Beacon frame or through active probing (Probe Response frame).

2) An optional Open System authentication may be invoked between the initiator and the peer STA.

3) The initiator and the peer STA will execute the Unicast Key Negotiation and Multicast Key Announcement after the BK is derived from a PSK.

4) The WPI algorithm uses the negotiated keys and cipher suites to protect the data.

If the WAPI security mechanism is negotiated to enable during (re)association between the STA and AP /STA, the mutual authentication and key negotiation shall be executed. Based on a certificate, the entire authentication procedure should include the Open System authentication, WAI Certificate Authentication, Unicast Key Negotiation and Multicast Key/SATKey Announcement. Based on a PSK, the entire authentication procedure should include the Open System authentication, Unicast Key Negotiation and Multicast Key/STAKey Announcement.

The authentication messages between the STAs shall be transported by the WAPI protocol with EtherType 0x88B4, and those messages between the STAs and ASU are transported by the UDP socket with port number 3810.

A WAPI security association (see 8.1.2.1) starts when the STA's SME decides to establish a WAPI security network and completes when the MLME-SETPROTECTION.Request primitive is invoked. The time a security association takes to set up shall be less than the MIB variable gb15629dot11WAPIConfigSATimeout. If the security association is not setup within the MIB variable gb15629dot11WAPIConfigSATimeout, the Open System authentication will be released.

8.1.1 The structure of the authentication system

8.1.1.1 WAPI Systems and Ports

STAs that attach to the WAPI security network, referred to in this part of ISO/IEC 8802 as WAPI Systems, have two kinds of logical channels to access the WAPI security network, referred to in this part of ISO/IEC 8802 as WAPI Ports. The WAPI Port consists of a WAPI Controlled Port and a WAPI Uncontrolled Port.

A non-AP STA or an AP provides the WAPI Uncontrolled Port for other STAs to exchange protocol information with the ASU. The non-AP STA or the AP also provides the WAPI Controlled Port which shall block data sending and receiving until completion of a successful authentication.

8.1.1.2 Controlled and Uncontrolled access

The access control of the AE and ASUE is based on the WAPI Port, shown as Figure 8.

The WAPI Uncontrolled Port allows the transmission of the authentication information (WAPI protocol information) in WLANs. The transmission process is uncontrolled by the current authorization state. The general protocol information (except WAPI protocol information) will exchange via the WAPI Controlled Port only if the authorization state is “authorized”. The WAPI Controlled and Uncontrolled Ports may be two logical ports of the same physical port of attachment to the WLAN; any frame received on the physical port is made available at both the WAPI Controlled and Uncontrolled Ports, subject to authorization state associated with the WAPI Port.

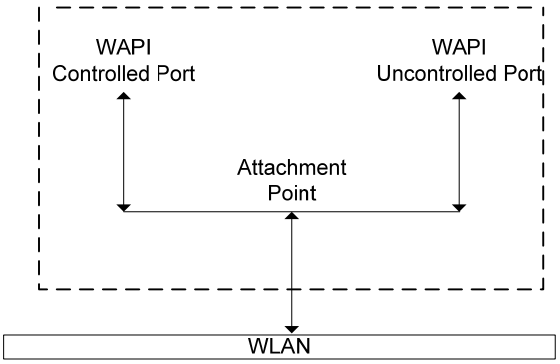


Figure 8 — WAPI Uncontrolled and Controlled Ports

Figure 9 depicts two different authorization states associated with the WAPI Controlled Port, representing the states of a switch that can be turned on or off. When the “switch” is turned off, the WAPI Controlled Port is unauthorized and is therefore disabled and prevents the flow of PDUs via that port. When the “switch” is turned on, the WAPI Controlled Port is authorized and is therefore enabled and allows the flow of PDUs via that port. There are two WAPI systems in Figure 9. In WAPI system 1, the authorization state of the WAPI Controlled Port is unauthorized (the “switch” is turned off). In WAPI system 2, the authorization state is “authorized” (the “switch” is turned on).

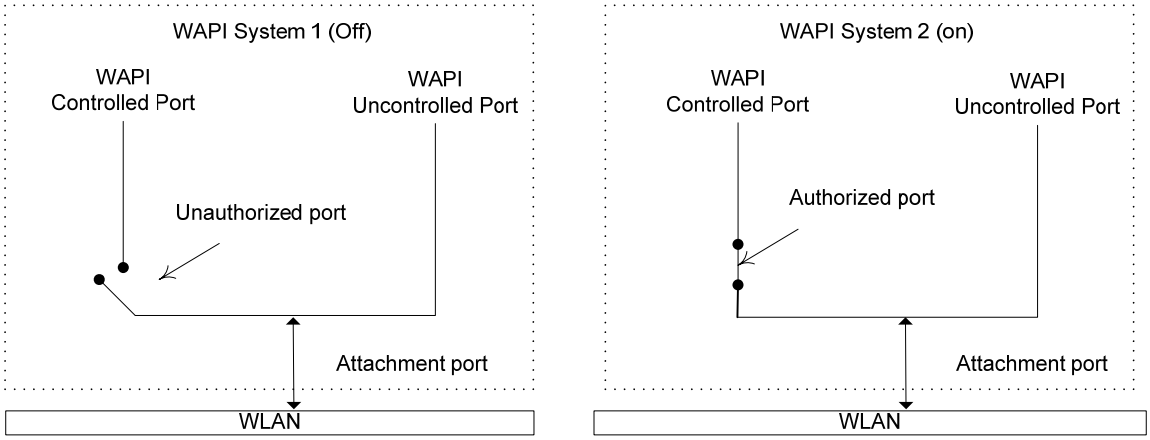


Figure 9 — Authorization state on WAPI Controlled Port

In the WAPI System, the state of every WAPI Port is determined by the value of gb15629dot11wapiControlledAuthControl parameter which may be “Authentication Disabled” or “Authentication Enabled”. When the parameter is set to “Authentication Disabled”, the state of the WAPI Controlled Ports is “Authorized”. When parameter is set to “Authentication Enabled”, the authorization state of every WAPI Controlled Port is determined by the value of gb15629dot11wapiControlledPortControl:

a) ForceUnauthenticated: the state of a WAPI Controlled Port is forced to be unauthorized. That is, the WAPI Controlled Port is unauthorized unconditionally (i.e. no frame is permitted to exchange at this controlled port).

b) Auto: the state of a WAPI Controlled Port is set according to the result of the mutual authentication between the AE and ASUE (i.e. the frame can pass through the WAPI Controlled Port only if the authentication is successful).

In the WAPI System, the network protocol information except the WAPI protocol information between STAs is exchanged via the WAPI Controlled Ports. Figure 10 illustrates the logical structure of the WAPI Controlled Port and the Uncontrolled Port. The authorization state of the WAPI Controlled Port in the system is set by the AE according to the result of the authentication exchanges between ASUE and authentication service unit (ASU).

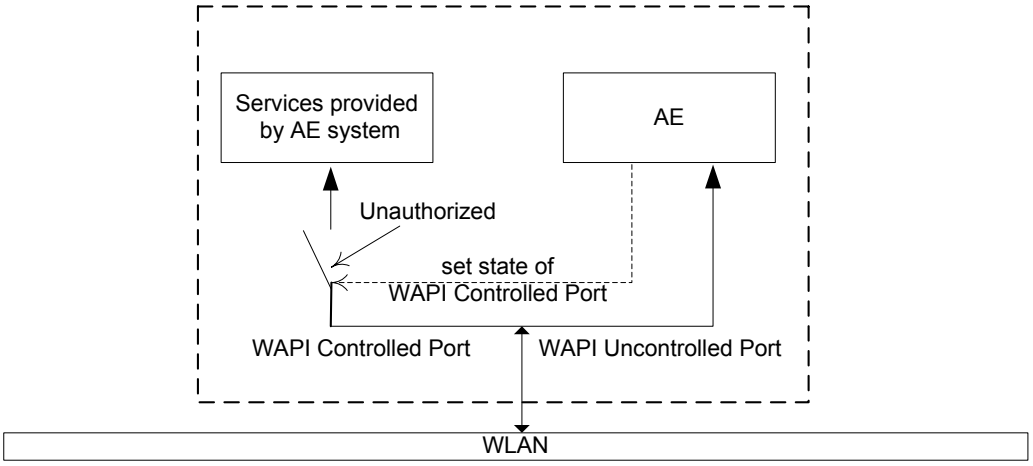


Figure 10 — Usage of the WAPI Controlled and Uncontrolled Port

Figure 11 illustrates the relationships and the exchange of information between the ASUE, AE and ASE. In this figure, both AE's and ASUE's Controlled Ports are in the unauthorized state and are therefore disabled from the point of view of access by the ASUE's request system to the services offered by the AE's system.

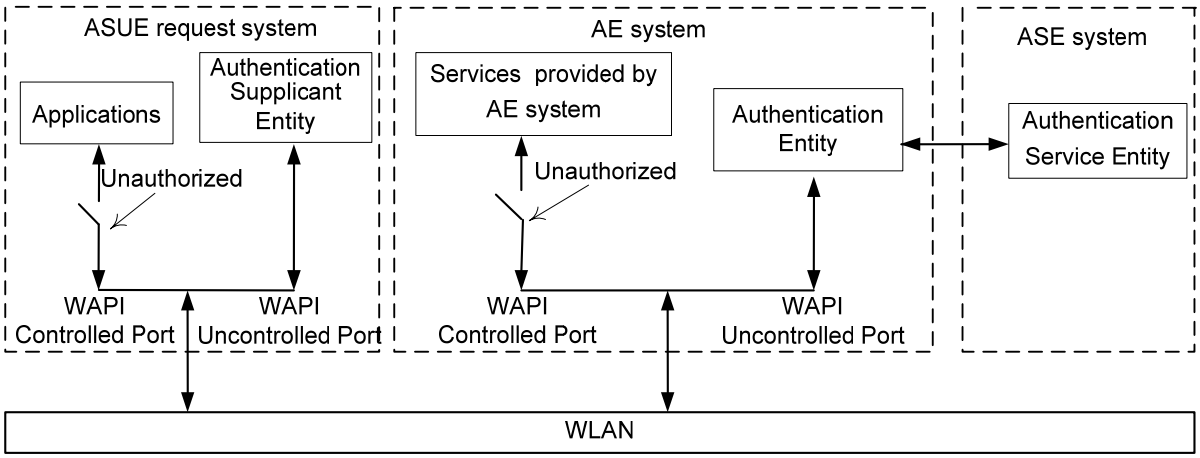


Figure 11 — Roles of ASUE, AE and ASE

8.1.2 WAPI security association management

8.1.2.1 WAPI security association definitions

A security association is a set of policy(ies) and key(s) used to protect information. There are four kinds of security associations supported by WAPI:

a) BKSA: The base key security association is from the PSK information or the result of a successful WAI Certificate Authentication (see 8.1.4.2);

a) USKSA: The unicast session key security association is the result of a successful Unicast Key Negotiation (see 8.1.4.3);

b) MSKSA: The multicast session key security association is the result of a successful Multicast Key Announcement (see 8.1.4.4);

c) STAKeySA: STAKey security association is the result of a successful STAKey Announcement (see 8.1.4.4).

8.1.2.1.1 BKSA

The BKSA is bidirectional. After a successful WAI Certificate Authentication or PSK configuration, BKSAs are established in both ASUE and AE. BKSAs are cached for up to their lifetimes and used to create the USKSA.

A BKSA consists of the following elements:

a) BKID: the identifier of BKSA;

b) AE MAC address;

c) ASUE MAC address;

d) BK;

e) lifetime;

f) AKMP;

g) other security parameters, depending on use (optional).

8.1.2.1.2 USKSA

The USKSA, the result of a successful Unicast Key Negotiation, is also bidirectional. The Unicast Key Negotiation is based on a BK, but the lifetime of the USKSA is independent of its superordinate BKSA. The USKSAs are cached for up to their lifetimes. For each ASUE and AE pair, there are two USKSAs at most. Generally, only one of the USKSAs is valid in a BSS. However, both of them may be valid during rekeying procedure. After the unicast MPDU encrypted by a new USKSA is received, the old USKSA is set to invalid. In an IBSS, if MAC_{AE}（the MAC address of the AE）> MAC_{ASUE}（the MAC address of the ASUE）, the negotiated USKSA is used to protect the unicast MPDU and the key update state is the same as in a BSS. If MAC_{AE} < MAC_{ASUE}, the negotiated USKSA shall be used in the Multicast Key Announcement, but not in encrypting unicast MPDUs. When a new USKSA is valid, the old one is set to invalid in the mean time.

The USKSA consists of the following elements:

a) USKID;

b) USK;

c) the selected unicast cipher suite;

d) lifetime;

e) AE MAC address;

f) ASUE MAC address;

g) other security parameters, depending on use, such as replay counter used in preauthentication and STAKey Establishment Request (Optional).

8.1.2.1.3 MSKSA

The MSKSA is the result of a successful Multicast Key Announcement. In a BSS, only one MSKSA is valid. It is used to encrypt broadcast/multicast MPDUs that are transmitted by the AP and decrypt broadcast/multicast MPDUs that are received by the non-AP STAs. In an IBSS, each STA has several valid MSKSAs, one is used for sending encrypted broadcast/multicast MPDUs and others are used for decrypting broadcast/multicast MPDUs sent by corresponding peer STAs. During the MSKSA update process, for every AP or peer STA, a STA has two valid MSKSAs used to decrypt broadcast/multicast MPDUs; only after a broadcast/multicast MPDU decrypted by the new MSKSA is received, the old MSKSA is set to invalid. However, for the MSKSA used to encrypt broadcast/multicast MPDUs, only one is valid.

The MSKSA consists of the following elements:

a) direction vector (whether the MSK is used for transmit or receive);

a) MSKID;

b) the selected Multicast cipher suite;

c) lifetime;

d) MSK;

e) AE MAC address;

f) other security parameters depending on use (optional).

8.1.2.1.4 STAKeySA

The STAKeySA is the result of a successful STAKey Announcement. It is a unidirectional security association from the initiator to the peer STA.

The MSKSA consists of the following elements:

a) STAKeyID;

b) STAKey;

c) unicast cipher suite (adopts multicast cipher suite announced by the AP);

d) initiator STA MAC address;

e) peer STA MAC address;

f) other security parameters, depending on use (optional).

8.1.2.2 Security associations

8.1.2.2.1 Security association in an ESS

When WAPI is enabled, the STA is identified by the WAPI Parameter Set information element which is contained in the Beacon frame and the Probe Response frame. If there is no such information contained in the frame, the WAPI security mechanism is not adopted in that STA.

There are three ways to establish BKSA for a roaming non-AP STA in an ESS.

a) After association, the BKSA can be directly established via the PSK or negotiated through a successful WAI Certificate Authentication.

b) A STA can cache BKSAs for APs in the ESS to which it has previously performed a successful WAI Certificate Authentication. If a STA wishes to roam to an AP for which it has cached one or more BKSAs, it can include one or more BKIDs in the WAPI security parameter set of its (Re)Association Request frame. An AP whose AE has retained the valid BKSA corresponded with the BKID in the received (Re)Association Request frame can skip the WAI Certificate Authentication and directly invoke the Unicast Key Negotiation. If none of the BKIDs of the cached BKSAs matches any of the supplied BKIDs or there are no BKIDs in the WAPI security parameter set of its (Re)Association Request frame, then the BKSA shall be established by the PSK or a WAI Certificate Authentication between the STA and AP.

c) If a STA has completed the establishment of BKSA and USKSA with an AP in an ESS. It may establish BKSAs with other APs in the same ESS through a preauthentication before association. The BKSAs shared between the STA and the new APs will be cached after a successful preauthentication. Based on the BKSA, the USKSA and MSKSA can be established by the Unicast Key Negotiation and Multicast Key Announcement, respectively.

8.1.2.2.2 Security association in an IBSS

If WAPI is enabled in an IBSS, a security association should be established between each pair of STAs. Any pair of STAs may negotiate to use any unicast cipher suite they both support. Each STA shall include the list of the multicast cipher suite and unicast cipher suite in its Beacon frame and Probe Response frame. Two STAs may only establish a BKSA only if they have advertised a common subset of unicast cipher suites and support the multicast cipher suite advertised by the peer STA.

In an IBSS, if a STA wants to set up a security association with a peer STA, but does not know the peer's security policy, it shall first obtain the peer's security policy using a Probe Request frame. Two STAs negotiate unicast cipher suites through the Unicast Key Negotiation. The selected unicast cipher suite is contained in the WAPI Parameter Set information element of the Unicast Key Negotiation Response. However, the WAPI Parameter Set information element contained in the Unicast Key Negotiation Confirmation is the same as in the Beacon frame and Probe Response frame. Two STAs will use the unicast cipher suite specified in the Unicast Key Negotiation initiated by the STA (as AE) with the higher MAC address.

If the certificate-based authentication is negotiated between two STAs, each of them initiates its own WAI Certificate Authentication, Unicast Key Negotiation and Multicast Key Announcement. Two sets of BKSAs, USKSAs and MSKSAs are established. If the PSK-based authentication is negotiated between two STAs, a BK is derived from the PSK, then a BKSA is established. Based on the BKSA, each of them initiates its own Unicast Key Negotiation and Multicast Key Announcement to establish two USKSAs and MSKSAs.

8.1.2.2.3 Deletion of the WAPI security association

When the STA receives any of the primitives of Association, Reassociation, Disassociation, Authentication and Deauthentication, or if it believes that it has drifted out of radio range of a STA, some security associations will be deleted.

In a BSS, a non-AP STA shall delete the USKSA and MSKSA, and the AP shall delete the USKSA.

In an IBSS, the STA will delete the USKSA and receive the MSKSA.

If the lifetime of a security association has run out, or the state of a security association is set to invalid, the security association will be deleted. Based on its own management policy, a STA may set the state of some security associations to invalid.

8.1.2.3 Selection of the WAPI security policy

The selection of the WAPI mechanism is completed utilizing the association process. The STA performs the WAPI security policy selection by the WAPI Parameter Set information element included in the (re)association frame.

The STA identifies the WAPI security policy it supports by the WAPI Parameter Set information element included in the Beacon frame and Probe Response frame. The other STAs that want to associate with the initiator shall choose the proper WAPI security policy from the received frames, then identify the WAPI security policy they choose by the WAPI Parameter Set information element included in the Association Request frame. Thus, the selection of the WAPI security policy is completed. If the STAs that want to associate with the initiator do not support the advertised policy, the association request shall be refused.

The STA and AP establish a WAPI security network by the steps below:

a) the STA chooses AP by the SSID;

b) the STA and AP execute Open System authentication;

c) the STA and AP execute an association process to complete the selection of WAPI security policy. Based on the selection, the STA and AP execute the processes described in 8.1.4.

In an IBSS, the STA chooses the WAPI security policy from the WAPI Parameter Set information element contained in the Beacon frame or Probe Response frame sent by the peer STA. The policy selected includes AKM and multicast cipher suites, ex unicast cipher suite. The WAI authentication can initiate according to the method specified in the AKM suite in the WAPI security policy it chooses. The unicast cipher suite is negotiated in the Unicast Key Negotiation.

8.1.3 Certificate

The Authentication Service Unit (ASU) is an important component of WAI authentication infrastructure based on the public-key cryptography technology. Its fundamental function is to check the validity of the user certificate.

The User certificate based on public-key algorithms is an important section of the architecture of the WAI system. The certificate can be used to uniquely verify the identity of a network user via the corresponding private key. It is the digital identity credential of a network user.

This part of ISO/IEC 8802 supports two kinds of public-key infrastructure certificates: X.509 v3 and GBW. The AE, ASUE and ASU certificates are used for the signature verification.

8.1.3.1 X.509 v3 Certificate

The framework and management of X.509 v3 certificate are defined in ITU-T Recommendation X.509,Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks.

In the certificate specified in this part of ISO/IEC 8802, the signature algorithm shall be ECDSA-192 and the hash algorithm shall be SHA-256. The ECDSA algorithm is defined in ANSI X9.62-Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA). The SHA-256 algorithm is defined in FIPS 180-2, Secure Hash Standard (SHS), August 2002.

The public-key algorithm identifier, signature algorithm identifier and parameters of ECDSA are denoted by object identifiers (OIDs) as follows:

a) ECDSA: OID=1.2.840.10045.2.1 in subjectPublicKeyInfo Field;

(the usage of signature should be specified in the keyUsage Field)

b) ECDSA-192 based on SHA-256: OID=1.2.156.11235.1.1.1 in signature Field;

c) ECC parameters: OID=1.2.156.11235.1.1.2.1 in parameters Field.

All elements in the X.509 v3 Certificate shall be encoded as ASN.1 DER. The encoding type, file format and suffix name of X.509 v3 Certificate shall be base64binary, Privacy Enhanced Mail (PEM) and ".cer" respectively.

8.1.3.2 GBW Certificate

8.1.3.2.1 GBW Certificate structure

Figure 12 shows the GBW Certificate structure.

Version
Serial number
Issuer name
Validity period
Subject name
Subject public-key information
Extension
Signature algorithm
Signature of the certification authority (CA)

Figure 12 — GBW Certificate structure

The elements of the certificate structure are described as follows:

a) Version

This identifies the structure of the GBW Certificate, which effects what valid items can be specified in and extracted from the certificate.

b) Serial number

Each GBW Certificate issued by ASU is assigned a unique serial number. The certificate subject shall be uniquely determined by the Serial number and Issuer name.

c) Issuer name

This specifies the identity of the issuer that signed the GBW Certificate.

d) Validity period

This specifies the time interval during which the GBW Certificate is valid. It adopts Universal Time Coordinated (UTC) time format, which indicates the number of seconds since 0 o'clock of 01/01/1970.

e) Subject name

This specifies the identity whose public-key the GBW Certificate identifies.

f) Subject public-key information

This is the public-key information of the subject.

g) Extension

This is kept for associating additional attributes.

h) Signature algorithm

This specifies the algorithms (including the hash algorithm identifier and signature algorithm identifier) used by the issuer on signature of the GBW Certificate. Here, the Signature algorithm is based on the ellipse curve cipher (ECC) mechanism authorized by China State Cryptography Administration, using WLANs.

i) Signature of the CA

This is the certificate issuer (ASU)'s signature of all the items (excluding the Signature algorithm) included in the GBW Certificate.

8.1.3.2.2 GBW Certificate format

Figure 13 shows the format of the GBW certificate.

	Version	Serial Number	Issuer Name	Validity Period	Subject	Subject Public-key	Extension	Signature Algorithm	Signature to Certificate
Octets:	2	4	6-256	8	6-256	variable	1	2	variable

Figure 13 — GBW Certificate format

The fields of the GBW Certificate format are described below.

a) The Version field shall be 2 octets in length. It shall contain an integer value that indicates the version number of a GBW certificat and the current version number is 1.

b) The Serial Number field shall be 4 octets in length. It shall contain an integer value.

c) The Issuer Name field contains a Length subfield and a Content subfield. The Length subfield shall be 1 octet in length and indicate the number of octets in the Content subfield.

d) The Validity Period field shall be 8 octets in length. The first 4-octet indicates the beginning time and the last 4-octet indicates the ending time. Both the beginning time and the ending time indicate the number of seconds since 0 o'clock of 01/01/1970.

e) The Subject field contains a Length subfield and a Content subfield. The Length subfield shall be 1 octet in length and indicate the number of octets in the Content subfield.

f) The Subject Public-key field contains a Length subfield and a Content subfield. The Length subfield shall be 2 octets in length and indicate the number of octets in the Content subfield. The Content subfield contains 3 elements: Public-key Algorithm Identifier, Parameter and Public-key.

    1) The Public-key Algorithm Identifier shall be 1 octet in length. ECDSA-192 is present when the value is 1. Other values are reserved.

    2) The Parameter contains 3 elements: Parameter Identifier, Length and Content subfields. The Parameter Identifier is 1 octet in length. The Length field shall be 2 octets in length and indicate the number of octets in the Content subfield. The value of Parameter is defined as follows when the value of the Public-key Algorithm Identifier is 1:

        i) Parameter Identifier = 1: The values of Parameter Identifier shall be denoted by OIDs. The Length field indicates the number of octets of OIDs. The values of Content subfield are the content of OIDs. In this part of ISO/IEC 8802, the OID of ECC parameter is 1.2.156.11235.1.1.2.1 which is authorized by the China State Cryptography Administration. OIDs are represented in ASN.1 DER encoding.

        ii) Other values are reserved.

    3) The Public-key specifies the public-key information. It has a length of 49 octets when the Public-key Identifier is 1.

g) The Extension field contains an Extension Attribute Number subfield and Extension Attribute List subfield. The Extension Attribute Number subfield shall be 1 octet in length and indicate the number of extended attibutes. The Extension Attribute List subfield specifies each extended attribute represented in the (Type-length-value) TLV format (See Figure 14).

h) The Signature Algorithm field specifies which algorithm should be used on signature of the certificate by the issuer. It contains a Length subfield and a Content subfield. The Length subfield shall be 2 octets in length and indicate the number of octets in the Content subfield. The Content subfield contains 3 elements: Hash Algorithm Identifier (1 octet), Signature Algorithm Identifier (1 octet) and Parameter.

    1) The values of the Hash Algorithm Identifier are defined as below:

        — Value 1: SHA-256;

        — other values are reserved.

    2) The values of the Signature Algorithm Identifier are defined as below:

        — Value 1: ECDSA-192;

— other values are reserved.

3) Parameter indicates the parameter information of the signature algorithm. It contains 3 elements: Parameter Identifier, Length and Content subfields. The Parameter Identifier shall be 1 octet in length. The Length subfield shall be 2 octets in length and indicate the number of octets in the Content subfield. The values of Parameter indicate the ECC parameters as follows when the value of Signature Algorithm Identifier is 1:

— Parameter Identifier = 1: The values of Parameter Identifier shall be denoted by OIDs. The Length field indicates the number of octets of OIDs. The values of Content subfield are the content of OIDs. In this part of ISO/IEC 8802, the OID of ECC parameter is 1.2.156.11235.1.1.2.1 which is authorized by the China State Cryptography Administration. OIDs are represented in ASN.1 DER encoding;

— other values are reserved.

i) The Signature to Certificate field contains a Length subfield and a Value subfield. The Length subfield shall be 2 octets in length and indicate the number of octets in the Value subfield. The Value subfield includes the issuer's signature of all the fields described above (excluding the Signature algorithm field) in the certificate. The value represented as a sequence of octets in the Value subfield should be the conversion outcome of the signature content. The conversion method is based on the rules in *ECC and Parameters of the ECDSA and ECDH Cryptography in WLAN Products* authorized by the China State Cryptography Administration.

Each extended attribute in the Extension field shall be represented in the Type-length-value (TLV) format as shown in Figure 14.

	Type	Length	Value
Octets:	1	2	variable

Figure 14 — TLV format

a) The Type field shall be 1 octet in length and indicate the kind of field that this part of the extended attribute represents. The values of the Type field are defined as below:

— Value 1: Authority Key Identifier;

— other values are reserved.

a) The Length field shall be 2 octets in length and shall indicate the number of octets in the Value field.

b) The Value field contains the data for this part of the extended attribute.

When the value of the Type field is 1, the Value field includes the content of Authority Key Identifier to identify the public-key used to verify the signature generated using the corresponding private key. The Authority Key Identifier attribute contains a Subject field and a Serial Number field of the issuer certificate.

The management of the certificate issuing and revoking, as well as the communication between ASUs, is beyond the scope of this part of ISO/IEC 8802.

8.1.3.2.3 Issue format of the GBW Certificate

The default suffix of the issue certificate file is ".wcr". All the fields in a certificate file shall be encoded and stored using the big endian order (except special announcement).

The issue format of the GBW Certificate is shown in Figure 15.

	Certificate file Identifier	Version	Certificate File Length	Digest	Attribute1	Attribute2	Attribute3
Octets:	16	2	2	4-65539	4-65539	4-65539	4-65539

Figure 15 — Issue format of the GBW Certificate

a) The Certificate File Identifier field specifies the file identification of the GBW Certificate to check if the issued certificate is in the GBW format. The value of this field should be “WAI15629.11-2003” with the form of ASCII code (denoted in hex: 0x57 0x41 0x49 0x31 0x35 0x36 0x32 0x39 0x2E 0x31 0x31 0x2D 0x32 0x30 0x30 0x33).This field shall be 16 octets in length.

b) The Version field specifies which version of GBW format applies to this certificate, and its current value is 1. This field shall be 2 octets in length.

c) The Certificate File Length field indicates the number of octets in the Digest and all attribute fields. This field shall be 2 octets in length.

d) The Digest field specifies the digest, generating by message digest algorithms, of the fields behind the Digest. The format of the Digest field is shown in Figure 16

	Digest Algorithm Identifier	Digest Length	Digest Data
Octets:	2	2	0-65535

Figure 16 — Digest field

1) The Digest Algorithm Identifier subfield specifies the digest algorithm and shall be 2 octets in length. The values of the Digest Algorithm Identifier are defined as below:

   — Value 1: SHA-256 (the value of the Digest length is 32 and the Digest data subfield includes the 32-octet digest content in the mean time);

   — other values are reserved.

2) The Digest Length subfield specifies the number of octets in the Digest Data subfield and shall be 2 octets in length.

3) The Digest Data subfield specifies the content of the digest.

e) The format of the Attribute field is shown in Figure 17.

	Attribute Identifier	Attribute Length	Attribute Data
Octets:	2	2	0-65531

Figure 17 — Attribute field

1) The Attribute Identifier subfield specifies the type of attribute and shall be 2 octets in length. The value of the Attribute Identifier defined as below:

— Value 1: the content of Attribute Data subfield is the issuer certificate;

— Value 2: the content of Attribute Data subfield is the user certificate;

— Value 3: the content of Attribute Data subfield is the secret key corresponding to user certificate;

— other values are reserved.

2) The Attribute Length subfield specifies the number of octets in the Attribute Data subfield and shall be 2 octets in length.

3) The Attribute Data subfield specifies the content of the attribute.

8.1.4 WAI protocol

8.1.4.1 WAI protocol message format

All the fields in a WAI protocol message shall be encoded and sent using the big endian order (except special announcement).

The format of the WAI protocol message between the ASUE, AE and ASU is defined in Figure 18.

	Version	Type	Subtype	Reserved	Length	Message Sequence Number	Fragment Sequence Number	Flag	Data
Octets:	2	1	1	2	2	2	1	1	variable

Figure 18 — Format of WAI protocol message in WAI authentication

a) The Version field specifies the version of authentication infrastructure and shall be 2 octets in length. The current value is 1.

b) The Type field specifies the protocol type and shall be 1 octet in length. The value of the Type field is defined as below:

— Value 1: WAI protocol message.

— Other values are reserved.

c) The Subtype field shall be 1 octet in length. If the value of the Type field is not 1, the values of Subtype field are reserved; otherwise, if the value of the Type field is 1, the values of the Subtype field are defined as below:

— Value 1: Preauthentication Start;

— Value 2: STAKey Establishment Request;

— Value 3: WAI Authentication Activation;

— Value 4: Access WAI Authentication Request;

— Value 5: Access WAI Authentication Response;

— Value 6: Certificate Authentication Request;

— Value 7: Certificate Authentication Response;

— Value 8: Unicast Key Negotiation Request;

— Value 9: Unicast Key Negotiation Response;

— Value 10: Unicast Key Negotiation Confirmation;

— Value 11: Multicast Key/STAKey Announcement;

— Value 12: Multicast Key/STAKey Announcement Response;

— Other values are reserved.

d) The Reserved field shall be 2 octets in length and the default value is 0.

e) The Length field specifies the number of octets in all the fields of the WAI protocol message and shall be 2 octets in length.

f) The Message Sequence Number field specifies the sequence number of the protocol message and shall be 2 octets in length. The sequence number of the first message is 1, and that of the following is increased by degree of 1.

g) The Fragment Sequence Number field specifies the sequence number of the fragment and shall be 1 octet in length. The sequence number of the first fragment is 0, and that of the following is increased by degree of 1.

h) The Flag field shall be 1 octet in length. The first bit (Bit 0) of the field spesifies whether or not a fragment is following. This bit is set to 1 when a fragment of the current WAI message is following; otherwise, it is set to 0. The remaining 7 bits of the field are reserved.

i) The content of the Data field is determined by the values of the Type field and the Subtype field. Besides the fixed content, some optional attributes are also included in this field.

The Message sequence number, Fragment sequence number and Frag field are valid only in those WAI protocol messages between ASUE and AE.

The maximal size of the WAI protocol message is 65535 octets.

8.1.4.1.1 Fixed content in the data field of the WAI protocol message

The following data fields of the WAI protocol message have fixed content.

a) FLAG

This field shall be 1 octet in length and its format is as shown in Figure 19.

B0	B1	B2	B3	B4	B5	B6	B7
BK Rekeying	Preauthentication	Certificate Authentication Request	Optional Field	USK Rekeying	STAKey Negotiation	STAKey Revoking	Reserved

Figure 19 — FLAG

1) Bit 0: BK Rekeying: The values are defined as below:

— Value 1: this is a BK rekeying message;

— Value 0: this is not a BK rekeying message.

2) Bit 1: Preauthentication: The values are defined as below:

— Value 1: this is a preauthentication message;

— Value 0: this is not a preauthentication message.

3) Bit 2: Certificate Authentication Request: The values are defined as below:

— Value 1: this requires verification of the counterpart's certificate.

— Value 0: this does not require verification of the counterpart's certificate.

4) Bit 3: Optional Field: The values are defined as below:

— Value 1: there is optional field in the message;

— Value 0: there is no optional field in the message.

5) Bit 4: USK Rekeying: The values are defined as below:

— Value 1: this is a USK rekeying message;

— Value 0: this is not a USK rekeying message.

6) Bit 5: STAKey Negotiation: The values are defined as below:

— Value 1: this is a STAKey negotiation message;

— Value 0: this is not a STAKey negotiation message.

7) Bit 6: STAKey Revoking: The values are defined as below:

— Value 1: this is a STAKey revoeking message;

— Value 0: this is not a STAKey revoking message.

8) Bit 7: Reserved.

b) BKID

This field shall be 16 octets in length and the computing method is: BKID= KD_HMAC_SHA-256(BK, MAC_{AE} ||MAC_{ASUE}). Here, "||" denotes concatenation.

c) USKID

This field shall be 1 octet in length, in which Bit 0 is significant.

d) MSKID/STAKeyID

This field shall be 1 octet in length, in which Bit 0 is significant.

e) Result

This field shall be 1 octet in length. The value may be 0 (success) or other values (reason codes).

f) Nonce

This field shall be 32 octets in length.

g) Key Data

This field contains a Length subfield and a Content subfield. The Length subfield shall be 1 octet in length and indicates the number of octets in the Content subfield.

h) Key Announcement Identifier

This field shall be 16 octets in length and it shall contain an integer value.

i) Data Sequence Number

This field shall be 16 octets in length and it shall contain an integer value.

j) Certificate

The format of the Certificate field is shown in Figure 20.

	Certificate Identifier	Certificate Length	Certificate Data
Octets:	2	2	0-65535

Figure 20 — Certificate

1) The Certificate Identifier subfield specifies the type of the certificate and shall be 2 octets in length. The values of the Certificate Identifier are defined as below:

   — Value 1: X.509 v3 certificate in the Certificate Data subfield;

   — Value 2: GBW certificate in the Certificate Data subfield;

   — other values are reserved.

2) The Certificate Length subfield specifies the number of octets in the Certificate Data subfield and shall be 2 octets in length.

3) The Certificate Data subfield specifies the content of the certificate.

k) Identity

The format of the Identity field is shown in Figure 21.

	Identity Identifier	Identity Length	Identity Data
Octets:	2	2	0-65535

Figure 21 — Identity

1) The Identity Identifier subfield specifies the type of Identity and shall be 2 octets in length. The values of the Identity Identifier subfield are defined as below:

— Value 1: The Identity Data subfield contains three elements of X.509 v3 certificate: the Subject subfield, the Issuer subfield and the Serial Number subfield;

— Value 2: The Identity Data subfield contains three elements of GBW certificate: the Subject subfield, the Issuer subfield and the Serial Number subfield.

— other values are reserved.

2) The Identity Length subfield specifies the number of octets in the Identity Data subfield and shall be 2 octets in length.

3) The Identity Data subfield includes the following certificate information: Subject, Issuer Name and Serial Number (see Figure 22).

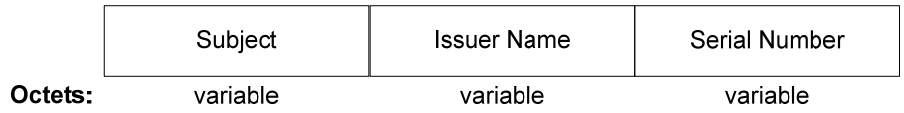


Figure 22 — Identity Data subfield

The identity data in X.509 v3 and GBW Certificate shall be encoded according to the rule of the field in which they reside.

l) ADDID

This field shall be 12 octets in length and its format is shown in Figure 23.

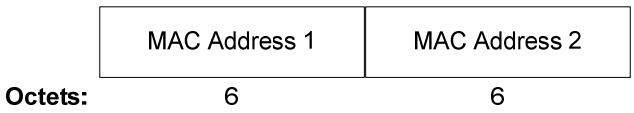


Figure 23 — ADDID

8.1.4.1.2 Attribute content in the data field of the WAI protocol message

8.1.4.1.2.1 General

The attribute shall be represented in the Type-length-value (TLV) format as shown in Figure 24.

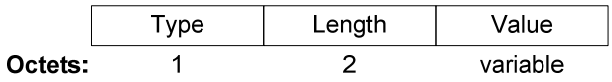


Figure 24 — Attribute format

a) The Type field specifies the type of the attribute and shall be 1 octet in length. The values of the Type field are defined as below:

— Value 1: Signature attribute;

— Value 2: Certificate authentication result;

— Value 3: Identity list;

— other values are reserved.

b) The Length field specifies the number of octets in the Value field and shall be 2 octets in length.

c) The Value field specifies the content of this attribute.

8.1.4.1.2.2 Value 1: Signature attribute

The format of the signature attribute is shown in Figure 25.

	Type=1	Length	Identity	Signature Algorithm	Signature Value
Octets:	1	2	variable	variable	variable

Figure 25 — Signature attribute

a) The Identity subfield: see section k) of 8.1.4.1.1.

b) The Signature Algorithm field contains two subfields: a Length subfield and a Content subfield. The Length subfield shall be 2 octets in length and indicates the number of octets in the Content subfield. The Content subfield contains three elements: Hash Algorithm Identifier (1 octet), Signature Algorithm Identifier (1 octet) and Parameter.

1) The values of the Hash Algorithm Identifier are defined as below:

— Value 1: SHA-256;

— other values are reserved.

2) The values of the Signature Algorithm Identifier are defined as below:

— Value 1: ECDSA-192;

— other values are reserved.

c) Parameter indicates the parameter information of the signature algorithm. It contains 3 elements: Parameter Identifier, Length and Content subfields. The Parameter Identifier shall be 1 octet in length. The Length subfield shall be 2 octets in length and indicate the number of octets in the Content subfield. The values of Parameter indicate the ECC parameters as follows when the value of Signature Algorithm Identifier is 1:

— Parameter Identifier = 1: The values of Parameter Identifier shall be denoted by OIDs. The Length field indicates the number of octets of OIDs. The values of Content subfield are the content of OIDs. In this part of ISO/IEC 8802, the OID of ECC parameter is 1.2.156.11235.1.1.2.1, which is authorized by the China State Cryptography Administration. OIDs are represented in ASN.1 DER encoding;

— other values are reserved.

The Signature Value field contains a Length subfield and a Value subfield. The Length subfield shall be 2 octets in length and indicates the number of octets in the Value subfield. The Value subfield includes the signature content. The value represented as a sequence of octets in the Value subfield should be the conversion outcome of the signature content. The conversion method is based on the rules in *ECC and*

Parameters of the ECDSA and ECDH Cryptography in WLAN Products authorized by the China State Cryptography Administration.

8.1.4.1.2.3 Value 2: Certificate Authentication Result

The format of the Attribute field is given in Figure 26.

Type=2 (1 octet)	Length (2 octets)
Nonce 1 (32 octets)	
Nonce 2 (32 octets)	
Authentication Result (1 octet)	Certificate1 (variable)
Authentication Result (1 octet)	Certificate2 (variable)

Figure 26 — Certificate Verification Result

8.1.4.1.2.4 Value 3: Identity List

The format of the Identity List field is given in Figure 27.

Type=3 (1 octet)	Length (2 octets)
Reserved (1 octet)	Number of Identities (2 octets)
Identity1 (variable)	
Identity2 (variable)	
…	

Figure 27 — Identity List

The Identity (1, 2, etc.) subfield: see 8.1.4.1.1 k).

8.1.4.2 WAI Certificate Authentication procedure

The WAI Certificate Authentication procedure is based on the certificate of the STA/AP to verify the identity and execute the key negotiation to establish BKSA. Figure 28 illustrates the procedure.

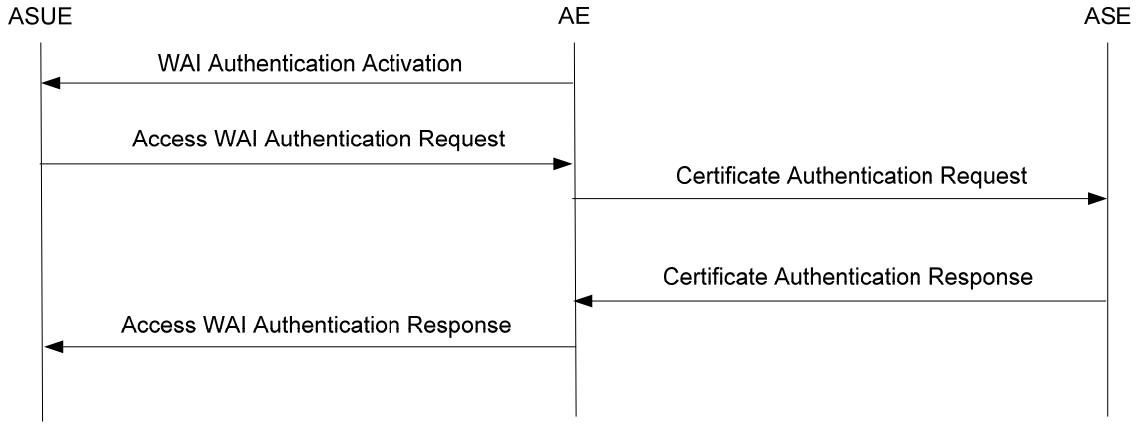


Figure 28 — WAI Certificate Authentication

8.1.4.2.1 WAI Authentication Activation

The format of the Data field of WAI Authentication Activation message (from AE to ASUE) is illustrated in Figure 29.

	FLAG	Authentication Identifier	Local ASU Identity	TA_{AE} Certificate	ECDH Parameter
Octets:	1	32	variable	variable	variable

Figure 29 —Format of the Data field of WAI Authentication Activation message

a) The FLAG field as defined in 8.1.4.1.1 a) shall be 1 octet in length. Only Bit 0 and Bit 1 are significant here. When the WAI Certificate Authentication procedure is invoked, Bit 0 (BK Rekeying) is set to 0; when the BK is refreshing in the WAI Certificate Authentication procedure, Bit 0 is set to 1. When the WAI Certificate Authentication is for Preauthentication, Bit 1 (Preauthentication) is set to 1; when the WAI Certificate Authentication is not for Preauthentication, Bit 1 is set to 0.

b) The Authentication Identifier field shall be 32 octets in length. If Bit 0 (BK Rekeying) is set to 0, the value of the Authentication Identifier field includes a random octet string generated by AE. If Bit 0 is set to 1, the value of the Authentication Identifier field shall be the same as the Authentication Identifier negotiated in the last WAI Certificate Authentication.

c) The Local ASU Identity field [see of 8.1.4.1.1 k)] specifies the ASU that the AE trusts.

d) The STA_{AE} (AE STA) Certificate field [See 8.1.4.1.1 j)] includes the certificate of a STA (as AE).

e) ECDH Parameter indicates the parameter information of the signature algorithm. It contains 3 elements: Parameter Identifier, Length and Content subfields. The Parameter Identifier shall be 1 octet in length. The Length subfield shall be 2 octets in length and indicate the number of octets in the Content subfield. The values of Parameter are defined as below:

   — Parameter Identifier = 1: The values of Parameter Identifier shall be denoted by OIDs. The Length field indicates the number of octets of OIDs. The values of Content subfield are the content of OIDs. In this part of ISO/IEC 8802, the OID of the ECC parameter is 1.2.156.11235.1.1.2.1, which is authorized by the China State Cryptography Administration. OIDs are represented in ASN.1 DER encoding.

   — other values are reserved.

The AE sends a WAI Authentication Activation message to ASUE to perform the mutual certificate authentication in the following situation:

a) the WAI-certificate-based AKM method is chosen in (re)association;

b) reauthentication based on a certificate is required according to the local policy of the AE;

c) the AE receives a Preauthentication Start message from the ASUE.

When the ASUE receives the WAI Authentication Activation message from the AE,

a) if the value of Bit 0 (BK Rekeying) of the FLAG field in the message is 1, the ASUE proceeds to Step b).Otherwise, it goes to Step c);

b) if the Authentication Identifier of the message does not match the one kept in the ASUE which is negotiated in the last WAI Certificate Authentication procedure, the ASUE discards the message; otherwise, the ASUE proceeds to Step c);

c) the ASUE chooses a certificate issued by the ASU which is included in the Local ASU Identity filed in the WAI Authentication Activation message, or according to ASUE's local policy. Then the ASUE generates temporary secret key x, temporary public-key $x \cdot P$ for the ECDH exchange and an ASUE challenge to construct the Access WAI Authentication Request message, sending it to the AE.

For the ECDH algorithm used in this part of ISO/IEC 8802, there are some explanations as follows.

a) Temporary private keys x and y are the integers between 1 and n-1, where n is the degree of the base point p in the elliptic curve domain parameters.

b) Temporary pubic keys p and $y \cdot P$ are the points in the elliptic curve defined in the elliptic curve domain parameters.

c) The key seed $(x \cdot y \cdot P)_{\text{abscissa}}$ negotiated by ECDH is the x-coordinate in the $x \cdot y \cdot P$. $x \cdot y \cdot P$ cannot be an infinite point.

8.1.4.2.2 Access WAI Authentication Request

The format of the Data field of Access WAI Authentication Request message (from ASUE to AE) is illustrated in Figure 30.

	FLAG	Authentication Identifier	ASUE Challenge	ASUE Key Data	STA_{AE} Identity	STA_{ASUE} Certificate	ECDH Parameter	ASU List trusted by ASUE	ASUE Signature
Octets:	1	32	32	variable	variable	variable	variable	variable	variable

Figure 30 — Format of the Data field of Access WAI Authentication Request message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. Here, only bits 0, 1, 2 and 3 are significant. The values in Bit 0 and Bit 1 should be the same as the values of FLAG field in the WAI Authentication Activation sent by the AE. The ASUE sets Bit 2 (Certificate Authentication Request) to 1 if it requires verification of the AE's certificate; otherwise, the ASUE sets Bit 2 to 0 if it does not require verification of the AE's certificate. Bit 2 shall be set to 1 if Bit 0 (BK Rekeying) is set to 0, i.e. if the message is not for BK Rekeying, the AE's certificate shall be verified. If there are some optional fields in the message, Bit 3 (Optional Field) is set to 1; otherwise, if there is no optional field in the message, Bit 3 (Optional Field) is set to 0.

b) The Authentication Identifier field shall be 32 octets in length. The value of the Authentication Identifier field shall be the same as the Authentication Identifier negotiated in the last WAI Certificate Authentication.

c) The value of the ASUE Challenge field (N_{ASUE}) is generated randomly by ASUE and shall be 32 octets in length.

d) The ASUE Key Data field [see 8.1.4.1.1 g)] specifies the ASUE's temporary public-key used in the ECDH exchange.

e) The STA_{AE} Identity field: see 8.1.4.1.1 k).

f) The STA_{ASUE} (ASUE STA) Certificate field [see 8.1.4.1.1 j)] specifies the certificate of a STA (as ASUE).

g) The ECDH Parameter field should be the same as the one in the WAI Authentication Activation message. When the ASUE initiates a BK Rekeying procedure, the value of ECDH Parameter field shall be the same

as the one in the initial WAI Authentication Activation message of the WAI Certificate Authentication procedure.

h) The ASU List trusted by ASUE field is optional and is presented as an Identity List defined in 8.1.4.1.2 c). The content of the field includes the ASUs trusted by the STA_{ASUE} except the issuer of the STA_{ASUE} certificate. If the ASUE trusts other entities besides its certificate issuer, this field can be used to inform the ASU about the case.

i) The ASUE Signature field is presented as a Signature attribute defined in 8.1.4.1.2 a). It is a signature to all the fields of the message excluding this field.

After receiving a WAI Authentication Activation message from the AE or when a BK rekeying is needed, the ASUE sends an Access WAI Authentication Request message to the AE.

When the AE receives the Access WAI Authentication Request message from the ASUE,

a) If the AE has not sent a WAI Authentication Activation message, then it checks whether the Authentication Identification field value of the Access WAI Authentication Request message matches the one negotiated in the last Certificate Authentication procedure. If so, the AE proceeds to Step b); otherwise the AE discards the message. If the AE has sent a WAI Authentication Activation message and the Authentication Identifier field value as well as Bit 0 and Bit 1 of the FLAG field match those in the WAI Authentication Activation message sent by the AE, respectively, the AE proceeds to Step b); otherwise, the AE discards the message.

b) In the following cases, the AE discards the message:

   — the value of the STA_{AE} Identity field does not match its own Identity;

   — the ECDH Parameter field dose not match the one in the WAI Authentication Activation message;

   — the signature of the ASUE is invalid.

   In the following cases, the AE constructs the Certificate Authentication Request message and sends it to the ASU; otherwise, in any other case, the AE proceeds to Step c):

   — the value of Bit 2 in the FLAG field is 1;

   — the AE requires utilizing the ASU to verify the STA_{ASUE} certificate according to its local applicaiton strategy.

c) If the STA_{ASUE} certificate which is verified by the AE (not ASU) is valid, the AE generates a 32-octet random number, named N_{AE} , as a challenge of AE, as well as the keys (a temporary private key y and a temporary public-key $y \cdot P$) used in the ECDH exchange. Then the AE obtains the master secret key seed $(x \cdot y \cdot P)_{abscissa}$ through the ECDH computation. By the function KD_HMAC_SHA-256 [$(x \cdot y \cdot P)_{abscissa}$ N_{AE}||N_{ASUE}|| “base key expansion for key and additional nonce”], the AE computes a 16-octet BK and a 32-octet Authentication Identifier seed. Then, by hashing the seed via the SHA-256 algorithm, a 32-octet Authentication Identifier value used in the next WAI Certificate Authentication procedure is obtained. Finally, the AE constructs the Access WAI Authentication Response message and sends it to the ASUE, in which the Access Result is set to success and the Bit 3 (Optional Field) of FLAG field is set to 0. On the other hand, if the ASUE’s certificate is invalid, the AE constructs the Access WAI Authentication Response message and sends it to the ASUE, in which the Access Result is set to failure, the Bit 3 (Optional Field) of FLAG field is set to 0, the AE Challenge field and the AE Key Data (AE’s temporary public-key) field are set to an arbitrary value. Finally, the AE invalidates an active association and Open System authentication with the STA_{ASUE}.

8.1.4.2.3 Certificate Authentication Request

The format of the Data field of Certificate Authentication Request message (from AE to ASU) is illustrated in Figure 31.

	ADDID	AE Challenge	ASUE Challenge	STA_{ASUE} Certificate	STA_{AE} Certificate	ASU List trusted by ASUE
Octets:	12	32	32	variable	variable	variable

Figure 31 — Format of the Data field of Certificate Authentication Request message

a) The ADDID field shall be assigned by $MAC_{AE}||MAC_{ASUE}$ (AE MAC address||ASUE MAC address) and 12 octets in length.

b) The AE Challenge field includes a random octet string generated by AE and shall be 32 octets in length.

c) The ASUE Challenge field shall be 32 octets in length and its value should be the same as the one in the Access WAI Authentication Request message.

d) The STA_{ASUE} Certificate field [see section k) of 8.1.1.4.1.1] value should be equal to the one in the Access WAI Authentication Request message.

e) The STA_{AE} Certificate field [see 8.2.1.4.1.1 k)] contains the certificate of the STA_{AE}.

f) The ASU List trusted by ASUE field is optional and its value should be equal to the one in the Access WAI Authentication Request message.

If the FLAG field in the Access WAI Authentication Request message indicates that a certificate needs to be verified or the AE itself needs to verify the certificate, the AE shall send a Certificate Authentication Request to the ASU.

After receiving an Access WAI Authentication Request message from the ASUE and sending a Certificate Authentication Request message to the ASU, the AE will not handle the Access WAI Authentication Request message sent by the ASUE until the Certificate Authentication Request message is timeout.

When the ASU receives the Certificate Authentication Request message,

a) The ASU verifies the STA_{AE} certificate and the STA_{ASUE} certificate. If neither certificate can be identified (e.g. the certificate issuer can not be found), the ASU sets both corresponding Authentication Results of the Certificate field in the Certificate Authentication Response message to ‘the issuer is unknown’, and then proceeds to Step b). If one certificate cannot be identified, the ASU sets the corresponding Authentication Result of the Certificate field in the Certificate Authentication Response message to ‘the issuer is unknown’ and the ASU verifies the state of the other certificate, then proceeds to Step b); otherwise, if both certificates can be identified, the ASU verifies the states of STA_{AE} and STA_{ASUE}, then proceeds to Step b).

b) According to the authentication results of the STA_{AE} certificate and STA_{ASUE} certificate, the ASU constructs a Certificate Authentication Response message with the signatures (Seriver Signature trusted by ASUE and AE values) and sends it to the AE.

8.1.4.2.4 Certificate Authentication Response

The format of the Data field of Certificate Authentication Response message (from ASU to AE) is illustrated in Figure 32.

	ADDID	Authentication Result of the Certificate	Server Signature trusted by ASUE	Server Signature trusted by AE
Octets:	12	variable	variable	variable

Figure 32 — Format of the Data field of Certificate Authentication Response message

a) The ADDID field shall be assigned by $MAC_{AE}||MAC_{ASUE}$ and 12 octets in length. Its value should be equal to the one in the Certificate Authentication Request message.

b) The Authentication Result of the Certificate filed includes the Certificate Authentication Result attribute as defined in 8.1.4.1.2 b). The values of Nonce1 and Nonce2 in the Certificate Authentication Result attribute are the same as the AE and ASUE Challenge values in the Certificate Authentication Request message respectively. The Certificate1 and Authentication Result1 elements in the Certificate Authentication Result attribute correspond to the STA_{ASUE} certificate. The Certificate2 and Authentication Rresult2 elements in the Certificate Authentication Result attribute correspond to the STA_{AE} certificate. The values of the Authentication Result of the Certificate filed are as follows:

- Value 0: the certificate is valid;
- Value 1: the certificate issuer is unknown;
- Value 2: the root certificate is untrusted;
- Value 3: the current time is not in the range of time for which the certificate is valid, or the time range of a certificate does not fall within the time range of the issuing certificate;
- Value 4: the certificate signature is invalid;
- Value 5: the certificate is revoked;
- Value 6: the certificate is not valid in its proposed usage;
- Value 7: the certificate revocation state is unknown;
- Value 8: the certificate has an unknown error;
- other values are reserved.

c) Server Signature trusted by the ASUE field includes the Signature Attribute as defined in 8.1.4.1.2 a). The value is a signature to the Authentication Result of the Certificate field.

d) Server Signature trusted by the AE field includes the Signature Attribute as defined in 8.1.4.1.2 a). The value is a signature to all the Data fields (except the ADDID and Server Signature trusted by AE) in this message.

If the ASE trusted by ASUE is the same as the one trusted by AE (the Identity attribute value in the Access WAI Authentication Request message is the same as the Identity attribute value in the WAI Authentication Activation message), then the Certificate Authentication Response message shall only include Server Signature trusted by ASUE field or Server Signature trusted by AE field. If the ASUE's Authentication Result attribute is set to 'the issuer is unknown', the Server Signature trusted by ASUE field shall not be included in the Certificate Authentication Response message.

After receiving a Certificate Authentication Request message from the AE, the ASUE will send the Certificate Authentication Response message to the AE.

When the AE receives the Certificate Authentication Response message,

a) according to the ADDID, the AE identifies the corresponding Certificate Authentication Request message. If the Nonce1 value in the Authentication Result of the Certificate field in the Certificate Authentication Response message matches the AE Challenge value in the Certificate Authentication Request message, proceeds to Step b); otherwise, the AE discards this message;

b) according to the value of Server Signature trusted by AE filed, the AE verifies the validity of the signature. If the signature is valid, proceeds to Step c); otherwise, the AE discards this message;

c) if the STA_{ASUE} certificate is valid, which is specified in the Certificate Authentication Response message by the ASU, the AE generates a 32-octet random number, named N_{AE} , as a challenge of AE, as well as the keys (a temporary private key y and a temporary public-key $y \cdot P$) used in the ECDH exchange. Then the AE obtains the master secret key seed $(x \cdot y \cdot P)_{abscissa}$ through the ECDH computation. By the formula KD_HMAC_SHA-256 [$(x \cdot y \cdot P)_{abscissa}$, $N_{AE}||N_{ASUE}||$ "base key expansion for key and additional nonce"], the AE computes a 16-octet BK and a 32-octet Authentication Identifier seed. Then, hashing the seed via the SHA-256 algorithm, a 32-octet Authentication Identifier value used in the next WAI Certificate Authentication procedure is obtained. Finally, the AE constructs the Access WAI Authentication Response message and sends it to the ASUE, in which the Access Result is set to success.

   On the other hand, If the STA_{ASUE} certificate is invalid, which is specified in the Certificate Authentication Response message by the ASU, the AE constructs the Access WAI Authentication Response message and sends it to the ASUE, in which the Access Result is set to failure. The AE Challenge field and the AE Key Data (AE's temporary public-key) field are set to an arbitrary value. Finally, the AE invalidates an active association and Open System authentication with the STA_{ASUE}.

   If the ASUE requires verification of the AE's certificate in the Certificate Authentication Request message, Bit 3 of FLAG in the Certificate Authentication Response message shall be set to 1; otherwise, Bit 3 of FLAG in the Certificate Authentication Response message shall be set to 0.

8.1.4.2.5 Access WAI Authentication Response

The format of the Data field of Access WAI Authentication Response message (from AE to ASUE) is illustrated in Figure 33.

	FLAG	ASUE Challenge	AE Challeng	Access Result	ASUE Key Data	AE Key Data	STA_{AE} Identity	STA_{ASUE} Identity	Multiple Certificate Verification Result	AE Signature
Octets:	1	32	32	1	variable	variable	variable	variable	variable	variable

Figure 33 — Format of the Data field of the Access WAI Authentication Response message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. Here only bits 0, 1 and 3 are significant. Bit 0 and Bit 1 should have the same value as the FLAG field in the Access WAI Authentication Request sent by the ASUE. If there are some optional fields in the message, AE sets Bit 3 (Optional Field) to 1; otherwise, if there is not optional field in the message, AE sets Bit 3 to 0.

   a) The ASUE Challenge field shall be 32 octets in length. It should be to the same as the one in the Access WAI Authentication Request message sent by the ASUE.

   b) The AE Challenge field is 32 octets in length. The value should be the same as the one in the Certificate Authentication Request message.

   c) The Access Result field shall be 1 octet in length. The values of the Access Result field are as follows:

— Value 0: success; the corresponding Authentication Result attribute value in the Certificate Authentication Response message is 0;

— Value 1: unidentified certificate; the corresponding Authentication Result attribute value in the Certificate Authentication Response message is 1;

— Value 2: certificate error; the corresponding Authentication Result attribute in the Certificate Authentication Response message are other values except 0 and 1;

— Value 3: prohibited by AE's local application strategy;

— other values are reserved.

b) The ASUE Key Data field [see 8.1.4.1.1 g)] includes the ASUE's temporary public-key used in the ECDH exchange. The field value should be the same as the one in the Access WAI Authentication Request message.

c) The AE Key Data field [see 8.1.4.1.1 g)] includes the AE's temporary public-key used in the ECDH exchange.

d) The STA_{AE} Identity field: see 8.1.4.1.1 k).

e) The STA_{ASUE} Identity field: see 8.1.4.1.1 k).

f) The Multiple Certificate Verification Result field is optional. It includes all the fields (except the ADDID) with the same content in the Certificate Authentication Response message.

g) The AE Signature field includes the Signature attribute as defined in 8.1.4.1.2 a). It is a signature to all the fields excluding itself.

After receiving the Certificate Authentication Response message or the Access Authentication Request message (both AE and ASUE do not require utilizing the ASU to verify the counterpart's certificate), the AE will send the Access WAI Authentication Response message to the ASUE.

When the ASUE receives the Access WAI Authentication Response message,

a) if the ASUE, according to STA_{AE} Identity and STA_{ASUE} Identity values in the Access WAI Authentication Response message, confirms that it corresponds to the current Access WAI Authentication Request message, the ASUE proceeds to Step b); otherwise, the ASUE discards the message;

b) if the values of Bit 0 and Bit 1 in the FLAG field match those in the Access WAI Authentication Request message respectively, the ASUE proceeds to Step c); otherwise, the ASUE discards the message;

c) if the ASUE Challenge and the ASUE Key Data values match those in the Access WAI Authentication Request message respectively, the ASUE proceeds to Step d); otherwise, the ASUE discards the message;

d) if the value of AE Signature is invalid, the ASUE discards the message; otherwise, the ASUE will verify the value of Access Result. If it is not "successful," the ASUE terminates the association and Open System authentication with the STA_{AE}; otherwise, the ASUE proceeds to Step e);

e) if the ASUE does not specify verification of the AE's certificate in the Access WAI Authentication Request message, the ASUE proceeds to Step f); otherwise, the ASUE will verify the trusted ASU's signature in the Multiple Certificate Authentication Result field. If the signature is invalid, the ASUE discards the message; otherwise, the ASUE will verify the value of Authentication Result of the Certificate. If it is invalid, the ASUE terminates the association and Open System authentication with the STA_{AE}; otherwise, the ASUE proceeds to Step f);

f) the ASUE obtains the master secret key seed $(x \cdot y \cdot P)_{abscissa}$ through the ECDH computation. By the formula KD_HMAC_SHA-256 [$(x \cdot y \cdot P)_{abscissa}$, $N_{AE}||N_{ASUE}||$ "base key expansion for key and additional nonce"], the ASUE computes a 16-octet BK and a 32-octet Authentication Identifier seed. Then, by hashing the seed via the SHA-256 algorithm, a 32-octet Authentication Identifier value used in the next WAI Certificate Authentication procedure is obtained.

8.1.4.3 Unicast Key Negotiation procedure

The Unicast Key Negotiation (see Figure 34) procedure uses the BK to complete the negotiation of the unicast session key (USK), and constructs the USKSA.

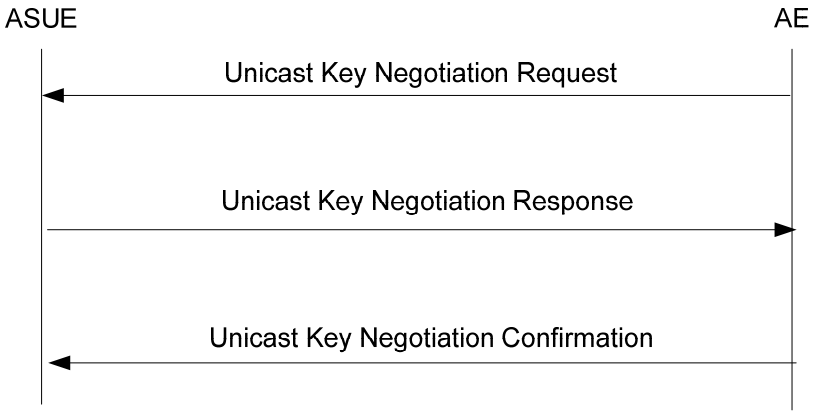


Figure 34 — Unicast Key Negotiation procedure

8.1.4.3.1 Unicast Key Negotiation Request

The format of the Data field of Unicast Key Negotiation Request message is illustrated in Figure 35.

	FLAG	BKID	USKID	ADDID	AE Challenge
Octets:	1	16	1	12	32

Figure 35 — Format of the Data field of Unicast Key Negotiation Request message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. Here, only Bit 4 (USK Rekeying) is significant. When the AE is refreshing the USK, Bit 4 is set to 1; otherwise, Bit 4 is set to 0.

b) The BKID field [see 8.1.4.1.1 b)] shall be 16 octets in length and indicates the index of the current shared BK.

c) The USKID field shall be 1 octet in length. Bit 0 of the field specifies the index of the current USK and the other bits are reserved. The default initial value for Bit 0 is 0 when the Unicast Key Negotiation is performed for the first time, and the value switches between 0 and 1 when a new Unicast Key Negotiation is performed.

d) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length. The MAC address 1 attribute value in the field is the AE's MAC address, and the MAC address 2 attribute value is the ASUE's MAC address.

e) The AE Challenge field shall be 32 octets in length, named N_1. If Bit 4 (USK Rekeying) is set to 0, N_1 is a random octet string generated by AE. If Bit 4 is set to 1, N_1 is to the same as the AE Challenge field value negotiated in the last Unicast Key Negotiation.

The AE will send a Unicast Key Negotiation Request message to the ASUE and start a Unicast Key Negotiation with the ASUE in the following cases:

a) when the AE successfully completes the WAI Certificate Authentication procedure and constructs the valid BKSA;

b) preshared key authentication method is enabled;

c) the cached BKSA is used;

d) the USK is refreshing.

After the ASUE receives the Unicast Key Negotiation Request message from the AE associated with this ASUE,

a) if the BKSA associated with a BKID is invalid, the ASUE discards the message; otherwise, the ASUE checks the value of Bit 4 of the FLAG field (USK Rekeying), if it is 0, the ASUE goes to Step c); if it is 1, the ASUE checks whether the USKSA associated with a USKID is valid, if so, the message, is discarded; otherwise, the ASUE proceeds to Step b);

b) if the AE Challenge is to the same as that negotiated in the last Unicast Key Negotiation, the ASUE proceeds to Step c); otherwise, the ASUE discards the message;

c) the ASUE generates a string of random octets as the ASUE Challenge value, named N_2. Then the ASUE generates 96 octets through the formula KD_HMAC_SHA-256 (BK, ADDID|| N_1||N_2||"pairwise key expansion for unicast and additional keys and nonce"), here, N_1 and N_2 are the values of AE and ASUE Challenge respectively, goes to Step d). The first 64 octets are the USK [the first 16 octets are the unicast encryption key (UEK), the second 16 octets are the unicast integrity check key (UCK), the third 16 octets are the message authentication key (MAK), and the fourth 16 octets are the Multicast Key/STAKey encryption key (KEK)]. The last 32 octets are the AE Challenge seed of the next Unicast Key Negotiation. By hashing the seed via the SHA-256 algorithm, a 32-octet AE Challenge value used in the next Unicast Key Negotiation is obtained;

d) the ASUE calculates the Message Authentication Code value via the HMAC_SHA-256 algorithm using the MAK and constructs the Unicast Key Negotiation Response message to send to the AE, goes to Step e);

e) the ASUE shall invoke the MLME-SETWPIKEYS.request primitive to configure the new USK when it is in a BSS. When the ASUE is in an IBSS, it shall invoke the MLME-SETWPIKEYS.request primitive to configure the new USK only if its MAC address is lower than the AE's MAC address. For the new USK, ASUE shall invoke the MLME-SETPROTECTION.request primitive only to initiate the receiving function, that is, the new USK shall only be used to decrypt the Data frame of the received unicast message.

8.1.4.3.2 Unicast Key Negotiation Response

The format of the Data field of Unicast Key Negotiation Response message is illustrated in Figure 36.

	FLAG	BKID	USKID	ADDID	ASUE Challenge	AE Challenge	WIE_{ASUE}	Message Authentication Code
Octets:	1	16	1	12	32	32	variable	20

Figure 36 — Format of the Data field of Unicast Key Negotiation Responding message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. Here, only Bit 4 (USK Rekeying) is significant. When the Unicast Key Negotiation is refreshing the USK, Bit 4 is set to 1; otherwise, Bit 4 is set to 0.

b) The BKID field [see 8.1.4.1.1 b)] shall be 16 octets in length and indicates the index of the current shared BK.

c) The USKID field [see 8.1.4.1.1 c)] shall be 1 octet in length. Bit 0 of the field specifies the index of the current USK and the other bits are reserved. The default initial value for Bit 0 is 0 when the Unicast Key Negotiation is performed for the first time, and the value switches between 0 and 1 when a new Unicast Key Negotiation is performed.

d) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length. The MAC address1 attribute value in the field is the AE's MAC address, and the MAC address2 attribute value is the ASUE's MAC address.

e) The AE Challenge field shall be 32 octets in length. If the ASUE initiates to refresh the USK and the USK Rekeying is set to 0, the value of the AE Challenge field is to the same as that in the Unicast Key Negotiation Request message. If the USK Rekeying is set to 1, the value of the AE Challenge field is to the same as that negotiated in the last Unicast Key Negotiation.

f) The ASUE Challenge field shall be 32 octets in length. The value shall be a string of random octets generated by the ASUE.

g) The WIE_{ASUE} field specifies the WAPI Parameter Set information element chosen by the ASUE's SME. In a BSS, the field value is the same as the WAPI Parameter Set information element sent by the Association Request frame from the ASUE. In an IBSS, the field contains the unicast cipher algorithm selected by the ASUE, the multicast cipher algorithm announced by the AE, and the currently used AKM suite list.

h) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the ASUE, utilizing the MAK specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code field in the WAI protocol message, excluding message header.

When the ASUE needs to refresh the USK, or receives the Unicast Key Negotiation Request message, the ASUE sends the Unicast Key Negotiation Response message to the AE.

When the AE receives the Unicast Key Negotiation Response message from the AE associated with this AE,

a) if the Bit 4 (USK Rekeying) of the FLAG field is 1, the AE proceeds to Step b); otherwise, the AE goes to Step c);

b) if there is a valid USKSA but the USKSA associated with a USKID in the message is invalid, the AE proceeds to Step c); otherwise, the AE discards the message;

c) if the AE Challenge value is correct, the AE proceeds to Step d); otherwise, the AE discards the message;

d) the AE generates 96 octets through the formula KD_HMAC_SHA-256 (BK, ADDID|| N_1||N_2||"pairwise key expansion for unicast and additional keys and nonce"), here, N_1 and N_2 are the values of AE and ASUE Challenge respectively. The first 64 octets are the USK [the first 16 octets are the unicast encryption key (UEK), the second 16 octets are the unicast integrity check key (UCK), the third 16 octets are the message authentication key (MAK), and the fourth 16 octets are the Multicast Key/STAKey encryption key (KEK)]. The last 32 octets are the AE Challenge seed of the next Unicast Key Negotiation. By hashing the seed via the SHA-256 algorithm, a 32-octet AE Challenge value used in the next Unicast Key Negotiation is obtained. The AE calculates the Message Authentication Code locally via the HMAC_SHA-256 algorithm using the MAK. If the calculated value matches that included in the message, the AE proceeds to Step e); otherwise, the AE discards the message;

e) if the Bit 4 (USK Rekeying) of the FLAG field is 1, the AE proceeds to Step f); otherwise, If the Bit 4 (USK Rekeying) of the FLAG field is 0, then:

— in a BSS, if the WIE_{ASUE} field value is to the same as the WAPI Parameter Set information element in the received Association Request frame, the AE proceeds to Step f); otherwise, the AE terminates the association and Open System authentication with STA_{ASUE};

— in an IBSS, if the unicast cipher algorithm in WIE_{ASUE} field is supported, the AE proceeds to Step f); otherwise, the AE terminates the association and Open System authentication with STA_{ASUE}.

f) the AE calculates the Message Authentication Code locally by the HMAC_SHA-256 algorithm, and sends the Unicast Key Negotiation Confirmation message to the ASUE, goes to Step g);

g) the AE shall invoke the MLME-SETWPIKEYS.request primitive to configure the new USK when it is in a BSS. When the AE is in an IBSS, it shall invoke the MLME-SETWPIKEYS.request primitive to configure the new USK only if its MAC address is higher than the ASUE's MAC address. For the new USK, ASUE shall invoke the MLME-SETPROTECTION.request primitive only to initiate the receiving-sending function, that is, the new USK shall be used to decrypt and encrypt the Data frame of unicast message. If the Unicast Key Negotiation is used to refresh the USK, the old USK shall be deleted once the new one is used to decrypt the message, or it shall be automatically deleted after 60 s since the new USK is used to encrypt and send the message for the first time.

8.1.4.3.3 Unicast Key Negotiation Confirmation

The format of the Data field of Unicast Key Negotiation Confirmation message is illustrated in Figure 37.

	FLAG	BKID	USKID	ADDID	ASUE Challenge	WIE_{AE}	Message Authentication Code
Octets:	1	16	1	12	32	variable	20

Figure 37 — Format of the Data field of Unicast Key Negotiation Confirmation message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length, and its value shall be the same as the one in the Unicast Key Negotiation Response message.

b) The BKID field [see 8.1.4.1.1 b)] shall be 16 octets in length, and its value shall be the same as the one in the Unicast Key Negotiation Response message.

c) The USKID field [see 8.1.4.1.1 c)] shall be 1 octet in length, and its value shall be the same as the one in the Unicast Key Negotiation Response message.

d) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length, and its value shall be the same as the one in the Unicast Key Negotiation Response message.

e) The ASUE Challenge field shall be 32 octets in length, and its value shall be the same as the one in the Unicast Key Negotiation Response message.

f) The WIE_{AE} field includes the WAPI Parameter Set information element sent by the AE in the Beacon frame and Probe Response frame.

g) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the AE, utilizing the MAK specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code element in the WAI protocol message, excluding message header.

After receiving the Unicast Key Response message, the AE will send the Unicast Key Negotiation Confirmation message to the ASUE.

When the ASUE receives the Unicast Key Negotiation Response message,

a) if the ASUE challenge value is to the same as the one in the Unicast Key Negotiation Response message, the ASUE proceeds to Step b); otherwise, the ASUE discards the message;

b) the ASUE calculates the Message Authentication Code locally via the HMAC_SHA-256 algorithm using the MAK. If the calculated value matches that included in the message, the ASUE proceeds to Step c); otherwise, the ASUE discards the message;

c) if the Bit 4 (USK Rekeying) of the FLAG field is set to 1, the AE proceeds to Step d); otherwise, if the Bit 4 (USK Rekeying) of the FLAG field set to is 0, the ASUE checks whether the WIE_{AE} field value is the same as the WAPI Parameter Set information element in the received Beacon and Probe Response frame, if so, the ASUE proceeds to Step d), otherwise, the ASUE terminates the association and Open System authentication with the STA_{AE};

d) the ASUE shall invoke the MLME-SETPROTECTION.request primitive to initiate the sending function, that is, the new USK shall be used to encrypt the Data frame of the unicast message. If the Unicast Key Negotiation is refreshing the USK, the old USK shall be deleted.

8.1.4.4 Multicast Key/STAKey Announcement procedure

The Multicast Key/STAKey Announcement procedure (see Figure 38) utilizes keys negotiated in the Unicast Key Negotiation to perform the Multicast Key/STAKey Announcement and establish the MSKSA or STAKeySA.

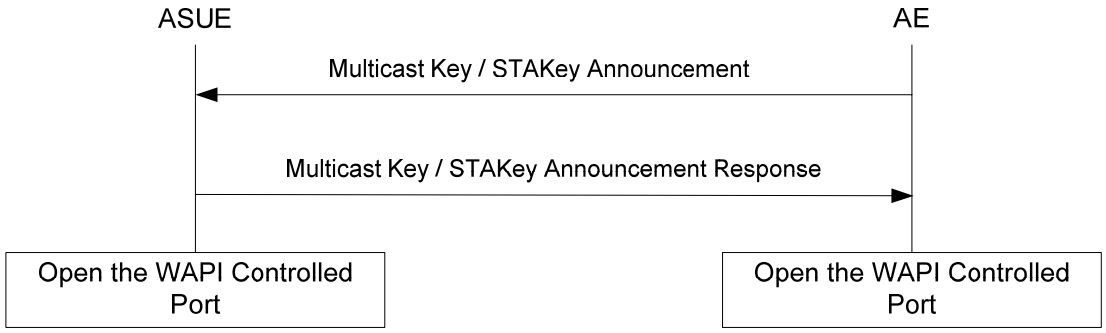


Figure 38 — Multicast Key/STAKey Announcement procedure

8.1.4.4.1 Multicast Key/STAKey Announcement

The AE should send ASUE the Multicast Key/STAKey Announcement message to announce the multicast /STAKey master key in the following situations:

a) after the success of the Unicast Key Negotiation;

b) when the AE needs to update the MSK;

c) when the AE receives the STAKey Establishment Request message.

The format of the Data field of Multicast Key/STAKey Announcement message is illustrated in Figure 39.

	FLAG	MSKID/ STAKeyID	USKID	ADDID	Data Sequence Number	Key Announcement Identifier	Key Data	Message Authentication Code
Octets:	1	1	1	12	16	16	variable	20

Figure 39 — Format of the Data field of Multicast Key/STAKey Announcement message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length and Bit 5 and Bit 6 are significant. When the key being announced is a STAKey, Bit 5 (STAKey negotiation) is set to 1; otherwise, it is set to 0. If Bit 6 (STAKey key deletion) is set to 1, it means to delete the STAKey determined by the USKID and ADDID; otherwise, it is set to 0.

b) The MSKID/ STAKeyID [see 8.1.4.1.1 d)] field shall be 1 octet in length. Bit 0 of the field indicates the index of the current key being announced and the other bits are reserved.The default initial value for Bit 0 is 0, and it will switch between 0 and 1 after each Multicast Key/STAKey Announcement.

c) The USKID field [see 8.1.4.1.1 c)] shall be 1 octet in length. Bit 0 of the field indicates the MAK used to calculate the value of the Message Authentication Code field.

d) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length. When the key being announced is a STAKey, the value of this field is: initiator MAC's address || peer MAC's address. When the key being announced is the MSK, the value of this field is $MAC_{AE}||MAC_{ASUE}$.

e) The Data Sequence Number field [see section i) of 8.1.4.1.1] shall be 16 octets in length. It shall contain an integer value and indicate the sequence number (PN in the WPI data message, see 8.2.3) of the data frame that has been encrypted and sent using the key being currently announced. Hereafter, the sequence number of the data frame that STA receives should be larger than the value of this field; otherwise, the frame shall be discarded.

f) The Key Announcement Identifier field [see 8.1.4.1.1 h)] shall be 16 octets in length. It shall contain an integer value (the field value defaulted to "0x5C365C365C365C365C365C365C365C36"). During each announcement procedure of a key updating, the value of this field will be increased by 1. If the key being announced is unchanged, the value of this field should remain unchanged. When the AE finds that the value of the Key Announcement Identifier field has overflowed, it shall terminate the Open System authentication with all associated STAs. The value of this field is also used as an initialization vector (IV) for creating a notification master key (NMK).

g) The Key Data field [see 8.1.4.1.1 g)] includes the NMK's cipher format, which is encrypted with the KEK by the negotiated unicast cipher algorithm. The NMK is a 16-octet random number generated by the AE, and the IV used in the encryption is the value of the Key Announcement Identifier field.

h) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the AE, utilizing the MAK key value specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code element in the WAI protocol message.

After the ASUE receives the Multicast Key/ STAKey Announcement message sent by the AE,

a) If Bit 5 of the FLAG field (STAKey Negotiation) is 1, the message is a STAKey Announcement message; otherwise, it is a Multicast Key Announcement message. If the ASUE does not support or not permit the use of STAKey, then when Bit 5 of the FLAG field (STAKey negotiation) is 1, this message is discarded. The ASUE uses the MAK specified in the USKID field to calculate a check value, and compares it with the Message Authentication Code field value. If they are the same, it goes to b); otherwise, this message is discarded.

b) The ASUE checks whether the value of the Key Announcement Identifier field increases monotonically. If so, it goes to c); otherwise, this message is discarded.

c) The ASUE gets a 16-octet NMK by decrypting the value of the Key Data. Then, through KD_HMAC_SHA-256 algorithm the ASUE generates a 32-octet session key (the first 16 octets are used as the encryption key, and the last 16 octets are used as the integrity check key), it goes to d). If Bit 5 of the FLAG field is 1, then the above NMK is the STAKey master key, and the session key is the STAKey session key; otherwise, the above keys are the multicast master key and MSK respectively.

d) The ASUE stores the value of the Key Announcement Identifier field, then constructs the Multicast Key/STAKey Response message and sends it to the AE, and then goes to e).

e) The ASUE configures or deletes the key according to the following cases:

— if this is a Multicast Key Announcement, the ASUE shall invoke the MLME-SETWPIKEYS.request primitive to configure the new MSK and invoke the MLME-SETPROTECTION.request primitive to initiate the receiving function, that is, the new MSK shall be used to decrypt the Data frame of the multicast message. If this Multicast Key Announcement, is performed for the first time, the WAPI Controlled Port shall be set to "On". If the Multicast Key Announcement is used for updating a MSK, the old one should be deleted when the Data frame of the multicast message is decrypted correctly using the new MSK;

— if this is a STAKey Announcement procedure, the initiator shall invoke the MLME-SETWPIKEYS.request primitive to configure the new STAKey and invoke the MLME-SETPROTECTION.request primitive to initiate the sending function. The peer shall configure the new STAKey and initiate the receiving function; that is, the new STAKey shall only be used to decrypt the Data frame of the received message. If the STAKey Announcement is used to update a STAKey, the old one should be deleted by the initiator immediately or by the peer when the Data frame of the STA-to-STA message is decrypted correctly using the new STAKey. If this is a STAKey deletion procedure, the peer will delete the corresponding STAKey.

If the AE announces a new MSK, the ASUE will store it. When the ASUE receives the multicast frame, it will choose the decryption key according to the KeyID field value. Once the ASUE receives the multicast message whose Data frame is encrypted using the new MSK by the AE, it checks whether the verification and decryption are correct. If so, the ASUE should discard the old MSK.

8.1.4.4.2 Multicast Key/STAKey Announcement Response

The format of the Data field of Multicast Key/STAKey Announcement Response message is illustrated in Figure 40.

	FLAG	MSKID/ STAKeyID	USKID	ADDID	Key Announcement Identifier	Message Authentication Code
Octets:	1	1	1	12	16	20

Figure 40 — Format of the Data field of the Multicast Key/STAKey Announcement Response message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. This field value should be the same as the one in the Multicast Key/ STAKey Announcement message sent by the AE.

b) The MSKID/ STAKeyID field [see 8.1.4.1.1 d)] field shall be 1 octet in length. This field value should be the same as the one in the Multicast Key/ STAKey Announcement message sent by the AE.

c) The USKID field [see 8.1.4.1.1 c)] shall be 1 octet in length. Bit 0 of the field indicates the MAK used to calculate the value of the Message Authentication Code field. This field value should be the same as the one in the Multicast Key/ STAKey Announcement message sent by the AE.

d) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length. This field value should be the same as the one in the Multicast Key/ STAKey Announcement message sent by the AE.

e) The Key Announcement Identifier field [see 8.1.4.1.1 h)] shall be 16 octets in length. This field value should be the same as the one in the Multicast Key/ STAKey Announcement message sent by the AE.

f) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the ASUE, utilizing the MAK specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code element in the WAI protocol message.

When the ASUE receives the Multicast Key/ STAKey Announcement message, it sends the Multicast Key/ STAKey Announcement response message to the AE.

After the AE receives the Multicast Key/ STAKey Announcement response message sent by ASUE, then:

a) the AE utilizes the MAK specified in the USKID field to calculate the check value, and then compares it with the value of the Message Authentication Code field. If they are the same, it goes to b); otherwise, this message is discarded;

b) the AE checks whether the values of the FLAG field, the MSKID/STAKeyID field, the USKID field, the ADDID field and the Key Announcement Identifier field are the same as those in the Multicast Key/STAKey Announcement message, respectively. If so, the Multicast Key/STAKey Announcement is successful, and it goes to c); otherwise, the AE discards this message;

c) the AE shall invoke the MLME-SETWPIKEYS.request primitive to configure the new key when the Multicast Key/STAKey Announcement is successful.

   — If this is a Multicast Key Announcement procedure, the AE shall invoke the MLME-SETPROTECTION.request primitive to initiate the sending function, that is, the new MSK shall be used to encrypt the Data frame of multicast message. If this Multicast Key Announcement is performed for the first time, the WAPI Controlled Port shall be set to "On". If the Multicast Key Announcement is used to update a MSK, the old one should be deleted when all ASUEs associated to the AE have been announced.

   — If this is a STAKey Announcement procedure, the AE shall directly forward the Data frame of the STA-to-STA message from the initiator to peer without encryption and decryption.

During the Multicast Key Announcement which is used to update a MSK, the AE uses the old MSK to encrypt the Data frame of multicast and send it to the ASUE. The AE will not use the new MSK being announced to the Data frame of multicast until all of the ASUEs associated with the AE have been successfully announced to the new key.

8.1.4.5 STAKey Establishment procedure

When a STA, e.g., named STA1, sends data to another STA, e.g., named STA2, in the same BSS, STA1 could require establishing STAKey with STA2 to encrypt the Data field of unicast message from STA1 to STA2. The encryption/decryption algorithm is announced by the AP in the Multicast Cipher Suite.

This STAKey Establishment procedure is based on the Multicast Key/ STAKey Announcement and is showed in Figure 41.

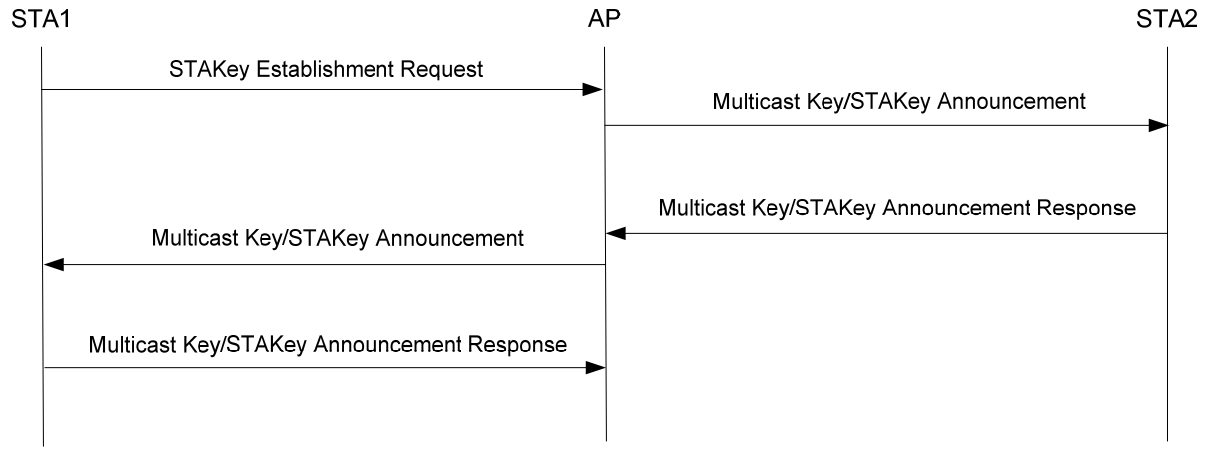


Figure 41 — Flow chart of the STAKey Establishment procedure

STA1 first sends the AP a STAKey Establishment Request. Then the AP uses the random number generator algorithm to generate the STAKey master key and transfers it to both STA1 and STA2 through the Multicast Key/ STAKey Announcement.

If the AP has announced the STAKey to STA2 successfully, but it is unsuccessful to STA1, then the AP should notify STA2 to delete the corresponding STAKey through the Multicast Key/ STAKey Announcement, and the AP should set the Bit 6 (STAKey Revoking) of FLAG to 1 in the Multicast Key/ STAKey Announcement message.

If STA1 wants to delete a certain STAKey, it should set the Bit 6 (STAKey Revoking) of FLAG to 1 in the STAKey Establishment Request message.

The format of the Data field of STAKey Establishment Request message is illustrated in Figure 42.

	FLAG	STAKeyID	USKID	ADDID	Replay Counter	Message Authentication Code
Octets:	1	1	1	12	16	20

Figure 42 — Format of the Data field in the STAKey Establishment Request message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length. Here, Bits 5 and 6 are significant, and Bit 5 (STAKey Negotiation) shall be set to 1. When the procedure is used to establish a STAKey, Bit 6 (STAKey Revoking) shall be set to 0; when the procedure is used to delete aSTAKey, Bit 6 shall be set to 1.

b) The STAKeyID field shall be 1 octet in length, and its value specifies the index of the SATKey chosen by the initiator.

c) The USKID field shall be 1 octet in length. Bit 0 of this field indicates the index of the MAK used to calculate the value of the Message Authentication Code field. The value of the USKID field should be the same as the one in the Multicast Key Announcement message sent by the AP.

d) The ADDID field shall be 12 octets in length, and its value specifies the index of the initiator's MAC address || the peer's MAC address.

e) The Replay Counter field shall be 16 octets in length. It shall contain an integer value (defaulted to 1). During each announcement procedure of a key updating, the value of this field will be increased by 1.

f) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the STA, utilizing the MAK specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code element in the WAI protocol message.

8.1.4.6 Preauthentication

If the targeted AP, e.g. named AP2, supports the Preauthentication service, the STA can execute the Preauthentication with the AP2 through the currently associated AP, e.g. named AP1. The STA requires to start the Preauthentication by sending a Preauthentication Start message to AP2. When AP1 receives the Preauthentication Start message, it shall check whether the Replay Counter and the Message Authentication Code values are valid. If so, it should forward the message to AP2; otherwise, the message will be discarded. When AP2 receives this Preauthentication Start message, it starts the WAI Certificate Authentication procedure by sending the WAI Authentication Activation message to the corresponding STA identified by the ADDID field value.

The format of the Data field of Preauthentication Start message is illustrated in Figure 43.

	FLAG	USKID	ADDID	Replay Counter	Message Authentication Code
Octets:	1	1	12	16	20

Figure 43 — Format of the Data field in the Preauthentication Start message

a) The FLAG field [see 8.1.4.1.1 a)] shall be 1 octet in length.

b) The USKID field [see 8.1.4.1.1 c)] shall be 1 octet in length. Bit 0 of this field indicates the index of the MAK used to calculate the value of the Message Authentication Code field. The value of the USKID field should be the same as the one in the Multicast Key Announcement message sent by the AP.

c) The ADDID field [see 8.1.4.1.1 l)] shall be 12 octets in length, and its value specifies the index of the initiator STA's MAC address || the destination AP's MAC address.

d) The Replay Counter field shall be 16 octets in length. It shall contain an integer value (defaulted to 1). For each time the STA sends the Preauthentication Start message, the value of this field will be increased by 1. This Replay Counter field value should be the same as the one in the STAKey Establishment Request message.

e) The Message Authentication Code field shall be 20 octets in length. Its value is generated by the STA, utilizing the MAK specified in the USKID field, through the HMAC_SHA-256 algorithm whose input values are the Data field values before this Message Authentication Code element in the WAI protocol message.

The STA can initiate the Preauthentication procedure with other APs such as AP2 only when the STA has successfully completed the Unicast Key Negotiation with the AP1 and configured the required USK. If the AP2 advertises the preauthentication capability in the WAPI Capability Information field of the WAPI Parameter Set information element, the STA can use the Preauthentication procedure with AP2. Only if the WAI Certificate Authentication and key management method are enabled, can the Preauthentication procedure be initiated.

To effect the preauthentication, the STA's ASUE sends a Preauthentication Start message with the DA being the BSSID of a targeted AP (AP2) and the RA being the BSSID of the AP with which it is associated (AP1). AP2 shall use a BSSID equal to the MAC address of its AE. AP2 should forward the Preauthentication messages sent by the STA to DS and by the DS to STA.

The AP2 that receives the Preauthentication Start message via the DS may initiate the WAI Certificate Authentication procedure to the STA via the DS. The DS should forward the message to AP1.

If the WAI Certificate Authentication completes successfully, the result of the Preauthentication may create a BKSA. If the STA has been associated to the preauthenticated AP2, the STA can use the BKSA with the Unicast Key Negotiation and Multicast Key Announcement. If the cached status of the Preauthentication result between the AP2 and STA is not synchronized, the Unicast Key Negotiation will fail. In this situation, the value of MIB attribute gb15629dot11WAPIStatsWAIUnicastHandshakeFailures will be increased by 1.

A STA's ASUE may initiate Preauthentication procedure with any AP within its present ESS with Preauthentication enabled regardless of whether the targeted AP is within radio range.

Even if a STA has preauthenticated, it is still possible that it may have to undergo a full WAI Certificate Authentication and the key negotiation procedure, as the AP's AE may have purged its BKSA due to, for example, unavailability of resources, delay in the STA associating, etc.

All WAI protocol messages (including all messages in the WAI Certificate Authentication, Unicast Key Negotiation, Multicast Key/STAKey Announcement and STAKey Establishment) are transferred by 0x88B4 protocol. If the format of a message is incorrect, the MIB attribute gb15629dot11WAPIStatsWAIFormatErrors value should be increased by 1. If the result of the signature verification is incorrect, the MIB attribute gb15629dot11WAPIStatsWAISignatureErrors value should be increased by 1. If the result of the MAC verification is incorrect, the MIB attribute gb15629dot11WAPIStatsWAIHMACErrors value should be increased by 1.

8.1.4.7 Cached BKSAs and WAPI key management

A STA can retain the BKSAs it establishes as a result of previous authentication. The BKSA cannot be changed while cached.

If a non-AP STA in an ESS has determined a valid BKSA with an AP to which is about to (re)associate, it includes the BKID for the BKSA in the WAPI Parameter Set information element of the (Re) Association Request. Upon receipt of a (Re)Association Request with one or more BKIDs, an AP checks whether its AE has cached a BKSA for the BKIDs and whether the BKSA is still valid. If so, it should initiate the Unicast Key Negotiation and Multicast Key Announcement; otherwise, it should initiate a full WAI authentication and key negotiation process after association has completed.

If both STA and AP assert possession of a cached BKSA, but the Unicast Key Negotiation is unsuccessful, they may delete the cached BKSA for the selected BKID.

If a STA roams to an AP with which it is preauthenticating and the STA does not have a BKSA for that AP, the STA shall initiate a full WAI Certificate Authentication and key negotiation procedure.

8.1.4.8 Rekeying

The AE can initiate the updating of the BK, USK and MSK, and the ASUE can initiate the updating of the BK and USK. In the key update procedure, the selected key index shall be currently invalid. For example, Bit 0 of the USKID field specifies the index of the current USK and the value of Bit 0 is 0 (value 1 is invalid); when a new Unicast Key Negotiation used to update a USK is performed, it shall set the Bit 0 to 1.The old key shall be deleted after the new one is active.

Once the BK is updated, the USK shall be updated as well. The process, from the beginning of initiating a BK update to the end of completing the USK update, shall be completed within the duration set by the value of the MIB attribute gb15629dot11WAPIConfigSATimeout, otherwise the two STAs will terminate the Open System authentication between them.

There is no confirmation of the WAPI Parameter Set information element or selection of the unicast cipher algorithm in the Unicast Key Negotiation used to update the USK. The STA still uses the unicast cipher algorithm negotiated in the initial Unicast Key Negotiation procedure.

8.1.4.9 Timeout processing

After the STA and AP send the WAI protocol message, if they do not receive any response within the timeout period, then increase the value of the MIB attribute gb15629dot11WAPIStatsWAITimeoutCounters by 1. The following occurs:

a) if this is a WAI Certificate Authentication message, then they re-send the message "gb15629dot11WAPIConfigCertificateUpdateCount" times;

b) if this is a Unicast Key Negotiation message, then they re-send the message "gb15629dot11WAPIConfigUnicastUpdateCount" times;

c) if this is a Multicast Key/ STAKey Announcement message, then they re-send the message "gb15629dot11WAPIConfigMulticastUpdateCount" times.

The default value of the timeout period of the WAI Certificate Authentication request message is set to 10 s, that of the Access WAI Authentication Request message is 31 s, and others are 1 s.

During the WAI Certificate Authentication, the Unicast Key Negotiation and Multicast Key Announcement, if there is still no response after re-sending the message a certain number of times (the number depending on the rules above), then the STA and AP will terminate the Open System authentication between them.

a) If this is a WAI Certificate Authentication procedure, then they increase the value of the MIB attributegb15629dot11WAPIStatsWAICertificateHandshakeFailures by 1.

b) If this is a Unicast Key Negotiation, then they increase the value of the MIB attribute gb15629dot11WAPIStatsWAIUnicastHandshakeFailures by 1.

c) If this is a Unicast Key Negotiation, then they increase the value of the MIB attribute gb15629dot11WAPIStatsWAIMulticastHandshakeFailures by 1.

During the STAKey Announcement procedure, if there is still no response after re-sending the message a certain number of times, then they increase the value of the MIB attribute gb15629dot11WAPIStatsWAIMulticastHandshakeFailures by 1.

8.1.4.10 WPI-SMS4 Key derivation architecture

In the WAI authentication and key management systems, the KD_HMAC_SHA-256 is used to derive each key and challenge from the corresponding master key.

8.1.4.10.1 Base Key derivation architecture

Base Key derivation architecture is shown as Figure 44.

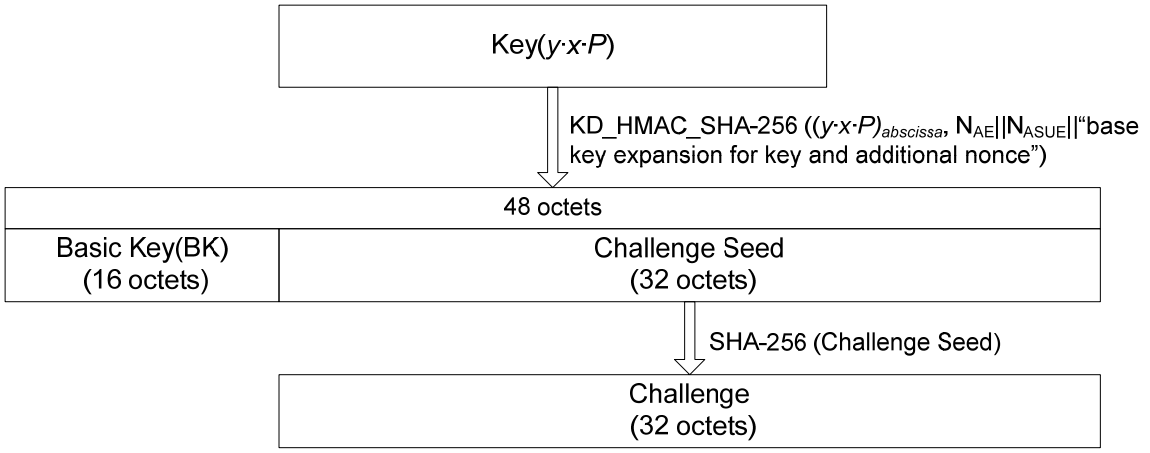


Figure 44 — Base key derivation architecture

8.1.4.10.2 Unicast key derivation architecture

Unicast Key derivation architecture is shown as Figure 45.

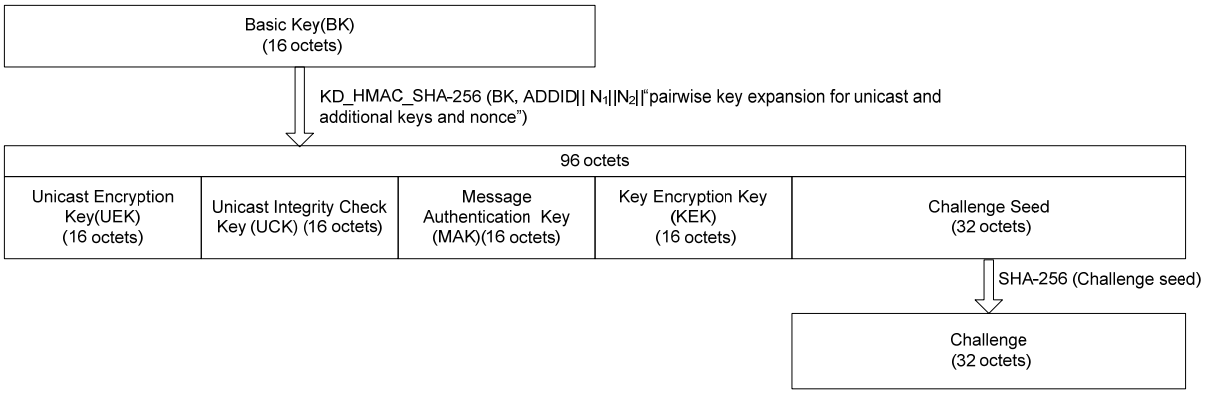


Figure 45 — Unicast key derivation architecture

8.1.4.10.3 Multicast/STAKey derivation architecture

Multicast/STAKey derivation architecture is shown as Figure 46.

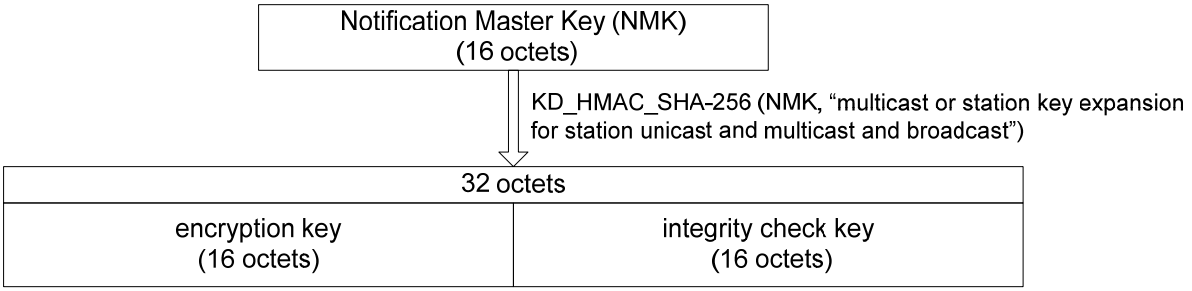


Figure 46 —Multicast/STAKey derivation architecture

8.1.4.10.4 Preshared key derivation architecture

Preshared key derivation architecture is shown as Figure 47.

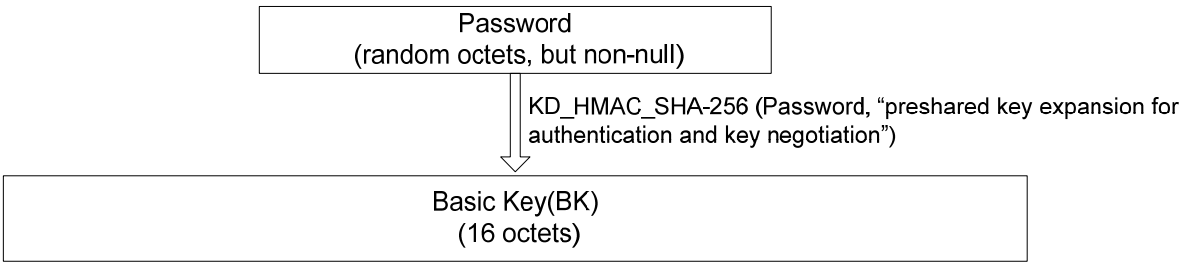


Figure 47 — Preshared key derivation architecture

8.1.4.11 WAI protocol message's fragmentation and defragmentation

When the STA or AP has a WAI protocol message to send, it needs to determine which local physical port to be used for sending the message, and then query this port to get the current Maximum Transmission Unit (MTU) size. Comparing the message length with the MTU size, if the message length is larger, the sender shall partition the message into smaller fragments. This is a fragmentation process. Each fragment shall have the same length (except the last one) and a separate WAI protocol header. The length should be a maximum value smaller than the MTU size. Here, we suggest this value be an integral multiple of 8 octets.

When these fragments reach the targeted receiver, it is necessary to reassemble them. The information contained in the WAI protocol message fragment header is enough for the receiver to correctly reassemble these fragments into a message.

Each fragment contains information to allow the complete message to be reassembled from its constituent fragments. The WAI protocol header of each fragment contains the information that is used by the destination STA to reassemble the message. This is a defragmentation process.

The WAI protocol layer utilizes three fields in the protocol message header to perform the fragmentation and defragmentation process. For each WAI protocol message from a sender, its Message Sequence Number field includes a unique value (the sequence number of the first WAI protocol message is 1, and that of the following is increased by a degree of 1), that value is copied into each fragment during the WAI protocol message fragmentation. If a message requires retransmission, its message sequence number remains unchanged. The Fragment Sequence Number field specifies the sequence number of the fragment (the sequence number of the first fragment is 0, and that of the following is increased by a degree of 1). Bit 0 of the FLAG field in the fragment specifies whether or not a fragment is following. Only the last fragment of the WAI protocol message shall have this bit set to 0; all other fragments of the WAI protocol message shall have this bit set to 1. Moreover, when a WAI protocol message fragmentation succeeds, the Length field value of each fragment denotes the size of the fragment, i.e. the total length of the WAI protocol header and the fragment data (in an octet).

Fragmentation and defragmentation should be completed in the WAI protocol layer, and the MAC layer should operate strictly according to the order in these processes. Messages shall be sent according to the message sequence number from small to large in the WAI protocol layer regardless of a non-AP STA or AP. When the fragmentation is enabled, the fragments of the same WAI protocol message shall be sent in order of the fragment sequence number. During the receiving process, if the WAI protocol layer of the destination STA has received all fragments of the same WAI protocol message, it will reassemble them to recover the original WAI protocol message, then process it. If a fragment of the WAI protocol message with incorrect sequence number is received, all the received fragments of the same WAI protocol message shall be discarded. If a WAI protocol message with incorrect sequence number is received, discard this WAI protocol message.

The timeout-based retransmission mechanism of each WAI protocol message has been already defined in the design of the WAI protocol exchange (including the WAI Certificate Authentication, Unicast Key Negotiation,

Multicast Key/STAKey Announcement and STAKey Establishment), therefore it is not necessary to specify that for each fragment, i.e. in a WAI protocol message exchange process, even if there is only one fragment lost, the whole WAI protocol message shall be retransmitted.

8.1.4.12 Port control and data transmission

When WAPI is enabled, the default authorization states of both the ASUE's and the AE's WAPI Controlled Ports are "*off*". When both the Unicast Key Negotiation and Multicast Key/STAKey Announcement are successfully completed, their WAPI Controlled Ports are set to "*on*".

During the updating of BKSA, USKSA and MSKSA, the authorization states of both the ASUE's and the AE's WAPI Controlled Ports are "*on*".

The WAI protocol message will be exchanged in plaintext format via the WAPI Uncontrolled Port regardless of the authorization state of the WAPI Controlled Port.

8.2 WPI privacy infrastructure

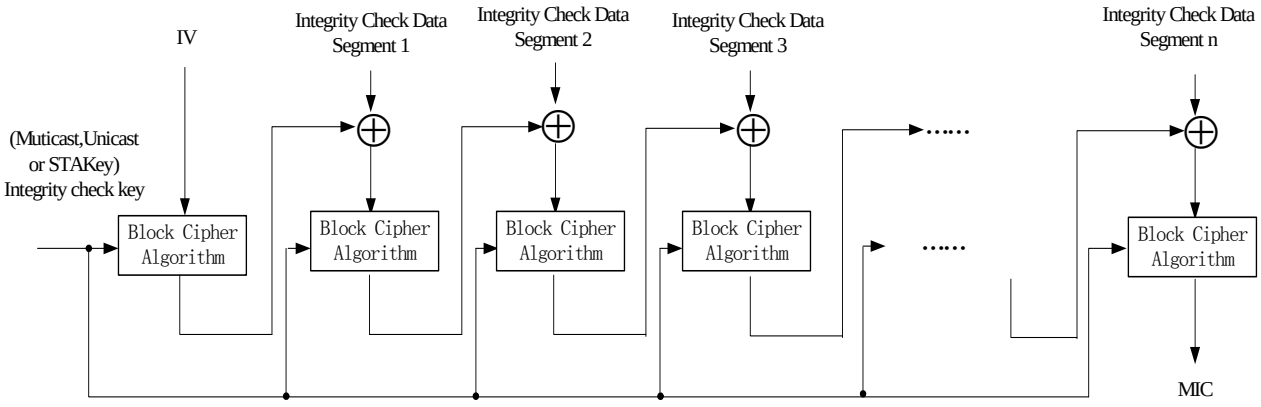
The WLAN Privacy Infrastructure (WPI) encrypts and decrypts the MPDUs (except the MPDUs of the WAI protocol message) in the MAC sub-layer. The WPI-SMS4 cipher suite selects the SMS4 block cipher algorithm. The modes of operation and encapsulation structure of the WPI-SMS4 are described in 8.2.1 to 8.2.4.

8.2.1 Modes of operation

In the WPI-SMS4 cipher suite, the integrity check algorithm works in CBC-MAC mode, and the data confidentiality uses the symmetric encryption algorithm in OFB mode. The CBC-MAC and OFB modes are as defined in ISO/IEC 10116:1997.

Two modes are illustrated in Figure 48.

CBC-MAC mode



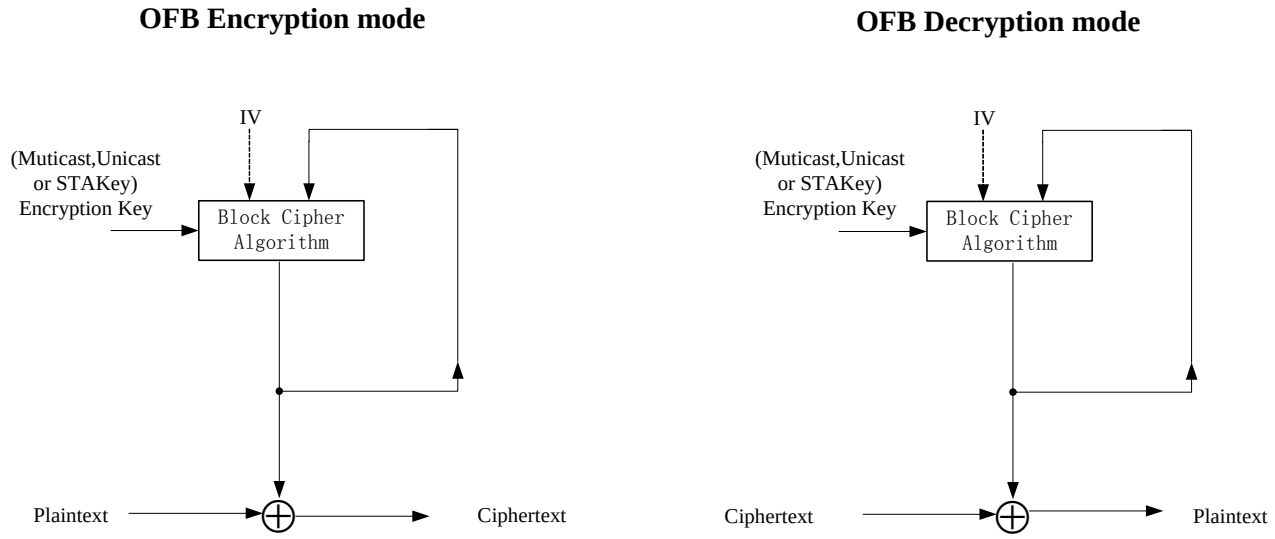


Figure 48 — Modes of operation

8.2.2 Key

8.2.2.1 Key management based on a certificate

During the WAI Certificate Authentication procedure, the ASUE and AE first use the ECDH exchange and the KD_HMAC_SHA-256 algorithm to negotiate a 16-octet BK, then during the Unicast Key Negotiation, the ASUE and AE respectively exchange a random octet string and use the KD_HMAC_SHA-256 algorithm and the BK to generate 96 octets. The first 64 octets are the USK [the first 16 octets are the unicast encryption key (UEK), the second 16 octets are the unicast integrity check key (UCK), the third 16 octets are the message authentication key (MAK), and the fourth 16 octets are the Multicast Key/STAKey encryption key (KEK)]. The last 32 octets are the AE Challenge seed of the next Unicast Key Negotiation. By hashing the seed via the SHA-256 algorithm, a 32-octet challenge value used in the next Unicast Key Negotiation is obtained. Finally, during the Multicast Key/STAKey Announcement, the ASUE and AE will respectively utilize the KD_HMAC_SHA-256 algorithm to extend the 16-octet NMK to generate a 32-octet MSK (the first 16 octets are used as the Multicast Key/STAKey encryption key, and the last 16 octets are used as the Multicast Key/STAKey integrity check key).

8.2.2.2 Key management based on the preshared key

In the preshared key mode, the ASUE and AE will use the shared key as the seed to generate the BK, then during the Unicast Key Negotiation, the ASUE and AE respectively exchange a random octet string and use the KD_HMAC_SHA-256 algorithm and the BK to generate 96 octets. The first 64 octets are the USK [the first 16 octets are the unicast encryption key (UEK), the second 16 octets are the unicast integrity check key (UCK), the third 16 octets are the message authentication key (MAK), and the fourth 16 octets are the Multicast Key/STAKey encryption key (KEK)]. The last 32 octets are the AE Challenge seed of the next Unicast Key Negotiation. By hashing the seed via the SHA-256 algorithm, a 32-octet challenge value used in the next Unicast Key Negotiation is obtained. At last, during the Multicast Key/STAKey Announcement, the ASUE and AE will respectively utilize the KD_HMAC_SHA-256 algorithm to extend the 16-octet NMK to generate a 32-octet MSK (the first 16 octets are used as the Multicast Key/STAKey encryption key, and the last 16 octets are used as the Multicast Key/STAKey integrity check key).

In a BSS or an IBSS, the key management can be based on the certificate or the preshared key. In an IBSS, the initiator and peer STAs respectively act as the AE and ASUE, and each of them is required to be the initiator to complete the Unicast Key Negotiation and Multicast Key Announcement. The USK negotiated in the

process initiated by the STA with a higher MAC address is chosen as the key which is used in the unicast transmission. The MAC address shall be 6 octets, taken to represent an unsigned binary number in big endian order for the purposes of comparison. When the user inputs the value of the preshared key, the support of input formats are both Hex and ASCII Character.

8.2.3 Encapsulation and decapsulation

8.2.3.1 General

The WPI's MPDU encapsulation structure is shown in Figure 49.

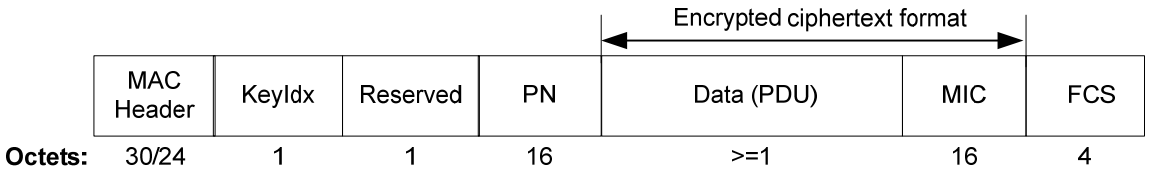


Figure 49 — WPI's MPDU encapsulation structure

a) The MAC Header field shall be 30 octets in length if the A4 (see Figure 50) exists; otherwise, it shall be 24 octets in length.

b) The KeyIdx field shall be 1 octet in length, and indicates the USKID, MSKID, or STAKeyID.

c) The Reserved field shall be 1 octet in length and the default value is 0.

d) The PN field shall be 16 octets in length. It shall contain an integer value, and indicates the sequence number of the data message. The PN value, which shall be encoded and sent using the big endian order, is used to encrypt and check the data as the IV in OFB and CBC-MAC modes.

e) The Data (PDU) field indicates the MPDU data, and its maximum length is 2278 octets [2312-18(WPI Header)-16(MIC)].

f) The MIC field shall be 16 octets in length.

g) The FCS field shall be 4 octets in length, and the value of this field is the MAC frame check sequence.

The MIC field value is calculated over the integrity check data using the integrity check key in CBC-MAC mode. The integrity check data includes two parts defined as below:

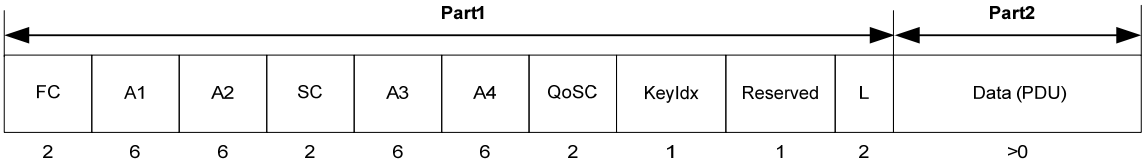


Figure 50 — Integrity check data

8.2.3.2 Part 1

Part 1 is described as follows.

a) Frame Control (FC) field (Bits 4, 5, 6, 11, 12 and 13 are set to 0; Bit 14 is set to 1) shall be 2 octets in length.

b) Address One (A1) field shall be 6 octets in length.

c) Address Two (A2) field shall be 6 octets in length.

d) Sequence Control (SC) field (Bits 4 to 15 are set to 0) shall be 2 octets in length.

e) Address Three (A3) field shall be 6 octets in length.

f) Address Four (A4) field shall be 6 octets in length. If A4 does not exist in the MAC header, the field value is set to 0.

g) QoS Control (QoSC) field may be 2 octets in length. If the MAC header contains the QoS Control field, this field exists.

h) KeyIdx field shall be one octet in length.

i) Reserved field shall be one octet in length.

j) L field shall be 2 octets in length and its value (in big endian order) specifies the length of the PDU data.

8.2.3.3 Part 2

Part 2 is described as follows.

a) Data (PDU) shall be one or more octets in length.

In the WPI-SMS4, when using CBC-MAC to calculate the MIC, the lenth of the integrity check data shall be a multiple of 16 octets. If the length of Part 1 or Part 2 is not a multiple of 16 octets, it shall pad the fewest number of “0” to make the length of Part 1 or Part 2 a multiple of 16 octets. The receiver will use the same method to verify.

During the data transmission, the WPI’s MPDU encapsulation procedure works as follows:

a) MIC can be calculated over the integrity check data by using the integrity check key and PN in CBC-MAC mode;

b) the ciphertext of “MPDU data || MIC” can be obtained by using the encryption key and PN to encrypt the “MPDU data || MIC” in OFB mode;

c) WPI encapsulates the MPDU and sends it.

During the data reception, the WPI’s MPDU decapsulation procedure works as follows:

a) it shall check whether the PN is valid. If not, then the MPDU is discarded, and the value of MIB attribute gb15629dot11WAPIStatsWPIReplayCounters will be increased by 1;

b) the received “MPDU data || MIC” ciphertext should be decrypted to recover the plaintext by using the decryption key and PN in OFB mode. If there is no valid decryption key, then the MPDU is discarded and the value of MIB attribute gb15629dot11WAPIStatsWPIDecryptableErrors will be increased by 1;

c) MIC can be recalculated over the integrity check data by using the integrity check key and PN in CBC-MAC mode. If the received and the locally computed MIC values are identical, the verification succeeds, and the MSDU shall be delivered to the upper layer. If the two differ, then the verification fails; the

receiver shall discard the MSDU and increase the value of MIB attribute gb15629dot11WAPIStatsWPIMICErrors by 1;

d) WPI decapsulates the MPDU and reassembles the plaintext.

8.2.4 Rules for using data message serial number PN

8.2.4.1 The unicast session

Once the USK is updated, the ASUE and AE initialize their PN values to “0x5C365C365C365C 365C365C365C365C36” and “0x5C365C365C365C365C365C365C365C37” respecively. Before a unicast frame is sent, the ASUE or AE shall first increase its PN value by 2.

When the ASUE receives a unicast frame from the AE, it should verify whether the value of PN corresponding to the USKID is an odd number and increases strictly monotonically. If not, the ASUE discards the message.

When the AE receives a unicast frame from the ASUE, it should verify whether the value of PN corresponding to the USKID is an even number and increases strictly monotonically. If not, the AE discards the message.

The AE can update the USK according to its strategy, such as time or amount of data packet. In addition, the overflow problem of a PN value can be solved through the USK update process initiated by the AE.

8.2.4.2 The multicast session

Once the MSK is updated, the AE initializes the value of PN to “0x5C365C365C365C365C365C365C365C36”. Before a multicast frame is sent, the AE shall first increase the value of PN by 1.

When the ASUE receives a multicast frame from the AE, it should verify whether the value of PN corresponding to the MSKID increases strictly monotonically. If not, the ASUE discards the message.

The AE can update the MSK according to its strategy such as time or amount of data packet. In addition, the overflow problem of a PN value can be solved through the MSK update process initiated by the AE.

8.2.4.3 The STA-to-STA session

Once a new STA-to-STA key is established, the initiator of the STAKey Negotiation procedure initializes the value of PN to “0x5C365C365C365C365C365C365C365C36”. Before a unicast frame is sent to the peer part, the initiator shall first increase the PN value by 1.

When the peer STA receives an encrypted unicast frame using the STAKey, it should verify whether the value of PN corresponding to STAKeyID increases strictly monotonically. If not, the peer discards the message.

The initiator can update the STAKey according to its strategy, such as time or amount of data packet. In addition, the overflow problem of a PN value can be solved by the STAKey update process initiated by the initiator.

8.3 WAPI Authentication and key management state machine

8.3.1 WAPI ASUE Authentication and key management state machine

The ASUE shall reinitialize the ASUE state machine whenever its system is initialized. An ASUE enters the AUTHENTICATION state on an event from the MAC that requests another STA to be authenticated. If the MIC or any of the WAI protocol messages fails, the ASUE silently discards the message. Figure 51 and Figure 52 depict the ASUE state machine.

The management entity will send an authentication request event when it needs to have an AE authenticated. This can be before or after the STA associates to the AP. In an IBSS environment, the event will be generated when a Probe Response frame is received.

Unconditional transfer (UCT) means the event triggers an immediate transition.

8.3.1.1 ASUE state machine states

The following list summarizes the states of the ASUE state machine:

a) DISCONNECT: A STA's ASUE enters this state when the WAI authentication procedure is timed out. The ASUE executes StaDisconnect and enters the INITIALIZE state.

b) INITIALIZE: A STA's ASUE enters this state from the DISCONNECT state, when it receives Disassociation or Deauthentication messages or when the STA initializes, causing the STA's ASUE to initialize the key state variables.

c) AUTHENTICATION: This state is entered when an Authentication Request message is sent from the management entity to authenticate a BSSID.

d) INITPSK: This state is entered when a PSK is configured.

e) BKNEGOTIATING: This state is entered when the WAI Authentication Activation message is received or if the BKReKey is set TRUE.

f) BKDONE: This state is entered when the Access WAI Authentication Response message is received and signature is verified.

g) USKNEGOTIATING: This state is entered from BKDONE to start the Unicast Key Negotiation when BKAvailable is set to TRUE and the Unicast Key Negotiation Request message is received or if the USKReKey is set to TRUE.

h) USKDONE: This state is entered when the Unicast Key Negotiation Confirmation message is received and MIC is verified.

i) MSKDONE: This state is entered when the Multicast Key Announcement message is received from the AE and MIC is verified.

j) IDLE: This state is entered when no Unicast Key Negotiation is occurring.

8.3.1.2 ASUE state machine variables

The following list summarizes the variables used by the ASUE state machine.

a) AuthenticationRequest: This variable is set to TRUE if the STA's ISO/IEC 8802-11 management entity requires an association to be authenticated. This can be set when the STA associates or at any other time.

b) DeauthenticationRequest: This variable is set to TRUE if a Disassociation or Deauthentication message is received.

c) Disconnect: This variable is set to TRUE when the STA should initiate a Deauthentication.

d) MessageReceived: This variable is set to TRUE when a WAI protocol message is received.

e) Init: This variable is used to initialize per-STA state machine.

f) TimeoutEvt: This variable is set to TRUE if the WAI protocol message sent out fails to obtain a response from the AE. The variable may be set by management action or set by the operation of a timeout while in the USKNEGOTIATING and BKNEGOTIATING states.

g) TimeoutCtr: This variable maintains the count of WAI protocol message received timeouts. It is incremented each time a timeout occurs on a WAI protocol message receive event and is initialized to 0.

h) HMACVerified: This variable is set to TRUE if the MIC on the received WAI protocol message is verified correct. Any WAI protocol message with an invalid HMAC will be dropped and ignored.

i) SIGVerified: This variable is set to TRUE if the signature to the received WAI protocol message is verified correct. Any WAI protocol message with an invalid signature will be dropped and ignored.

j) USK: This variable is the current USK.

k) MSK: This variable is the current MSK.

l) BK: This variable is the buffer holding the current BK.

m) BKReKey: This variable is set to TRUE when a BK rekeying is required.

n) USKReKey: This variable is set to TRUE when a USK rekeying is required.

o) Certificate_Verificaton: This variable is set to TRUE when the certificate is required to be verified.

p) PortControlled: This variable is the method by which the port is controlled. The value is Auto or ForceUnauthenticated.

q) PortEnabled: This variable is set to TRUE if the port control mechanism is applied.

r) PortState: This variable is set to "on" if the data frame is permitted to pass through the port.

s) Certificate Authentication: This variable is set to TRUE if the WAI certificate authentication and key management is chosen.

t) PSK Authentication: This variable is set to TRUE if the PSK authentication and key management is chosen.

8.3.1.3 ASUE state machine procedures

The following list summarizes the procedures used by the ASUE state machine:

a) STADisconnect (): Execution of this procedure deauthenticates the STA.

8.3.2 WAPI AE Authentication and key management state machine

There is one state diagram for the AE. In an ESS, the AE will always be on the AP; and in an IBSS environment, the AE will be on every STA.

The state diagram, shown in parts in Figures 53 through 57, consists of the following states:

a) The AUTHENTICATION, INITPSK, BKSTART, BKNEGOTIATING, BKNEGOTIATING2, BKNEGOTIATING3, BKDONE, USKSTART, USKNEGOTIATING, USKDONE, DISCONNECT, DISCONNECTED, and INITIALIZE states. These states handle the initialization, WAI Certificate Authentication, Unicast Key Negotiation, tear-down and general clean-up. These states are per associated STA.

b) The IDLE, MSKNEGOTIATING, KEYERROR, MSKDONE states. These states handle the transfer of the MSK to the associated client. These states are per associated STA.

c) The MSK_INIT, SETKEYS, and SETKEYSDONE states. These states change the MSK when required, trigger all the USK MSK state machines, and update the ISO/IEC 8802-11 MAC in the AE's AP when all STAs have updated the MSK. These states are global to the AE.

When the MSK is to be updated, the variable MSKReKey is set. The SETKEYS state updates the MSK and triggers all the USK MSK state machines that currently exist—one per associated STA. Each USK MSK state machine sends the MSK to its STA. When all the STAs have received the MSK (or failed to receive the key), the SETKEYSDONE state is executed which updates the APs encryption/integrity engine with the new key.

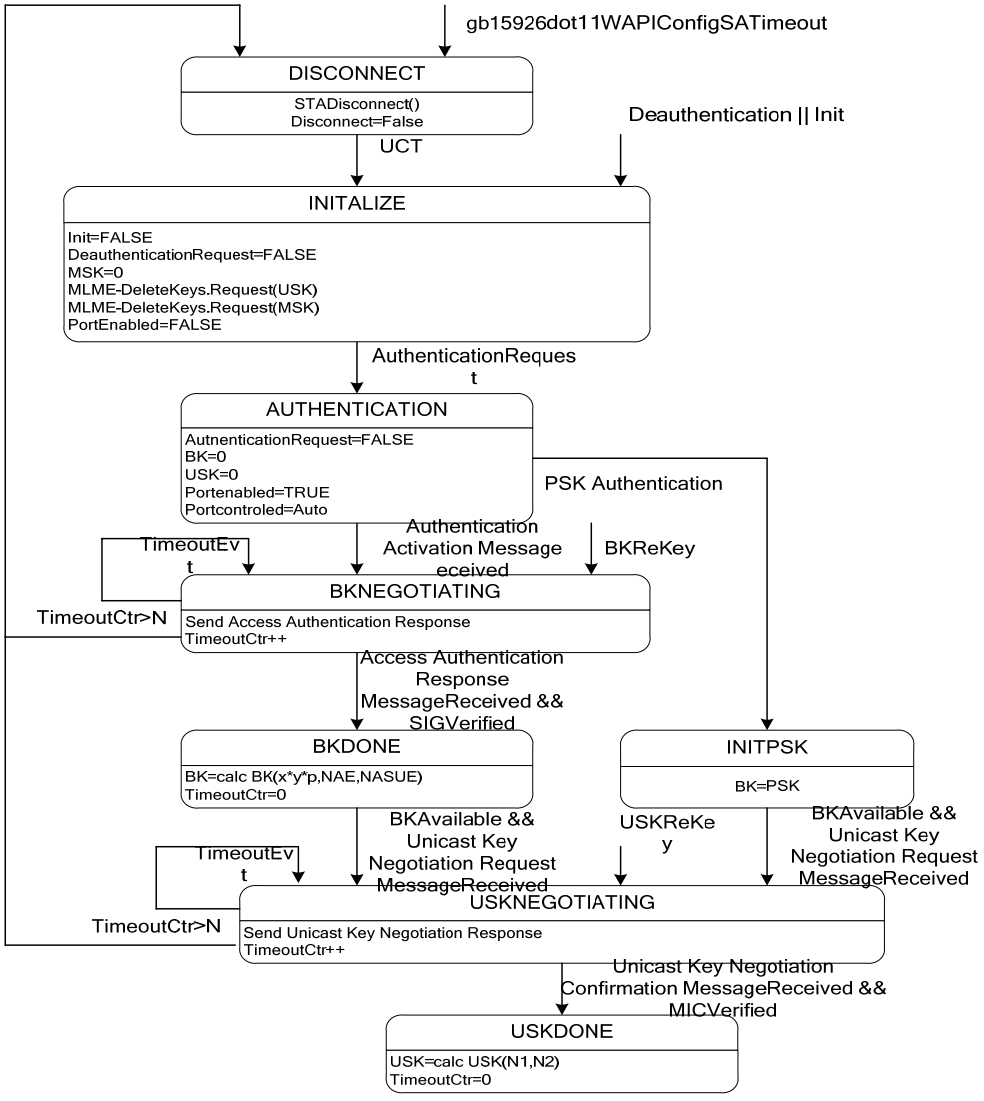


Figure 51 — ASUE state machine, Part 1

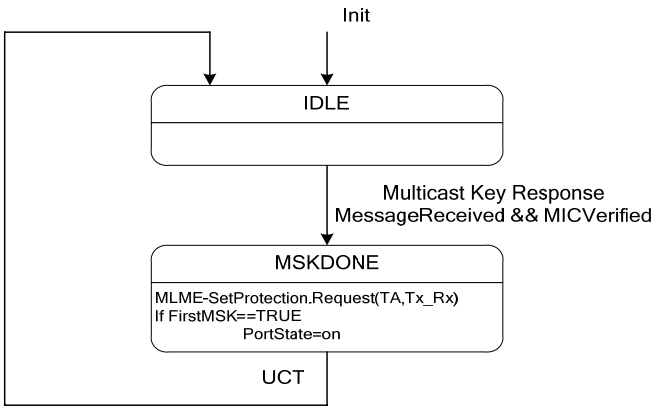


Figure 52 —ASUE state machine, Part 2

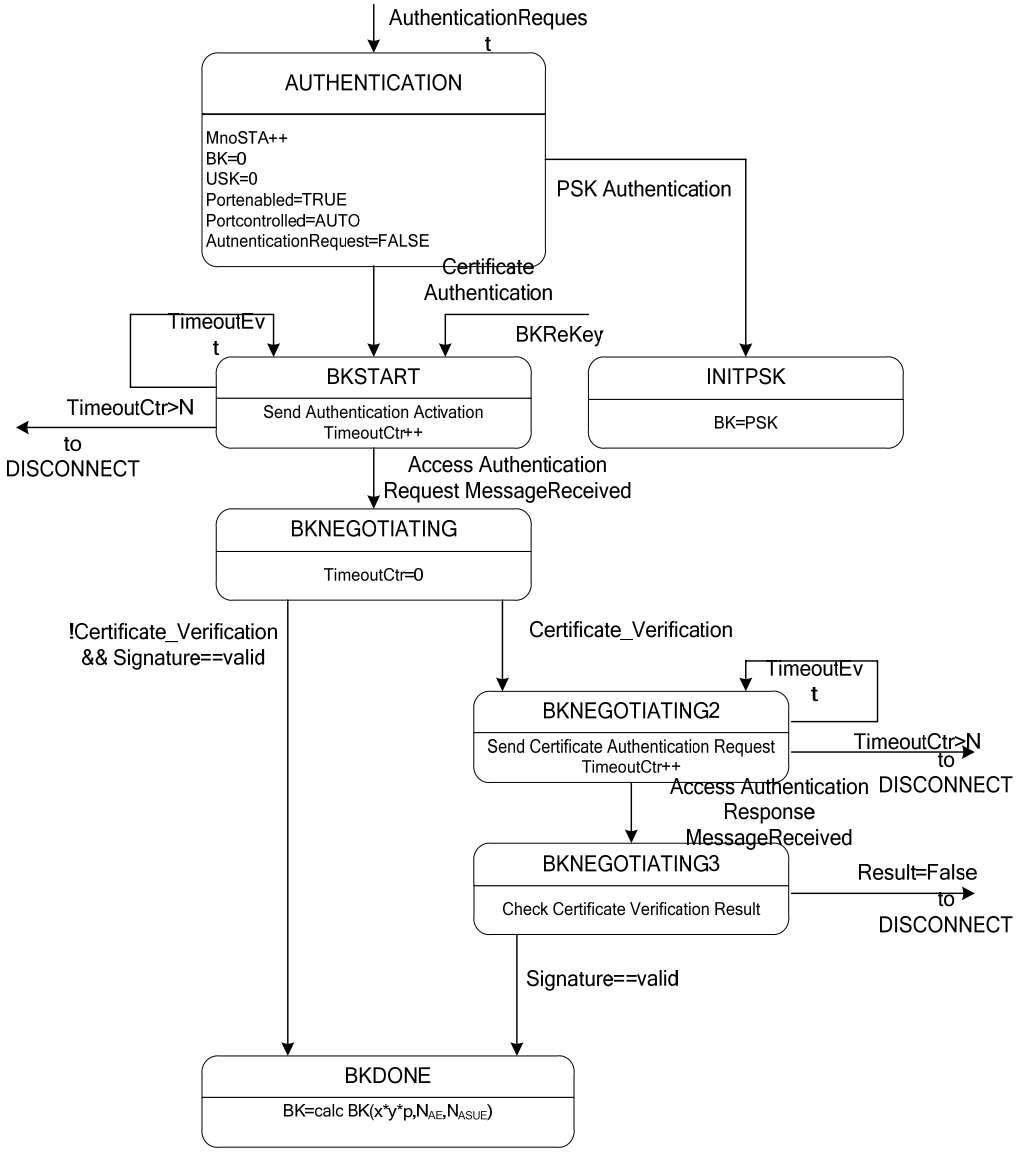


Figure 53 — AE state machine, Part 1

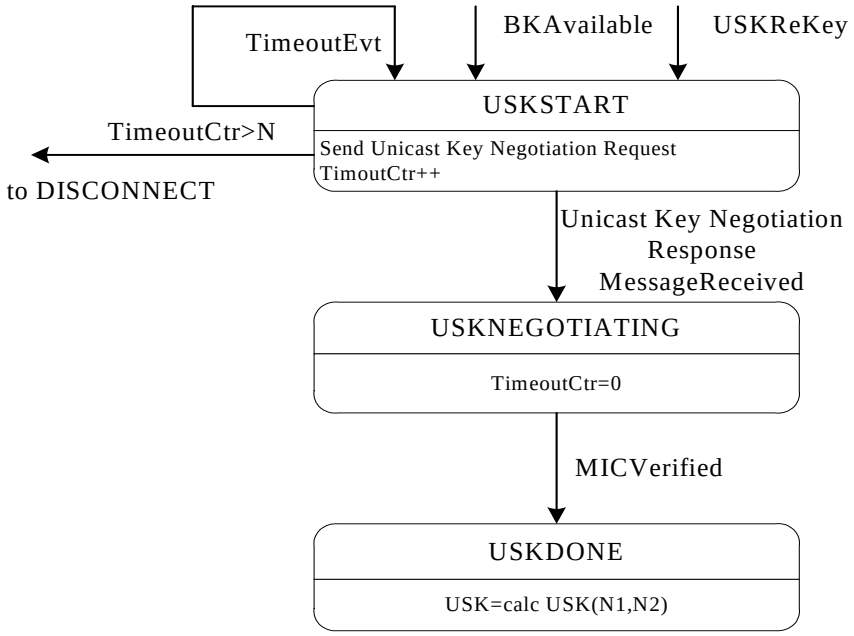


Figure 54 — AE state machine, Part 2

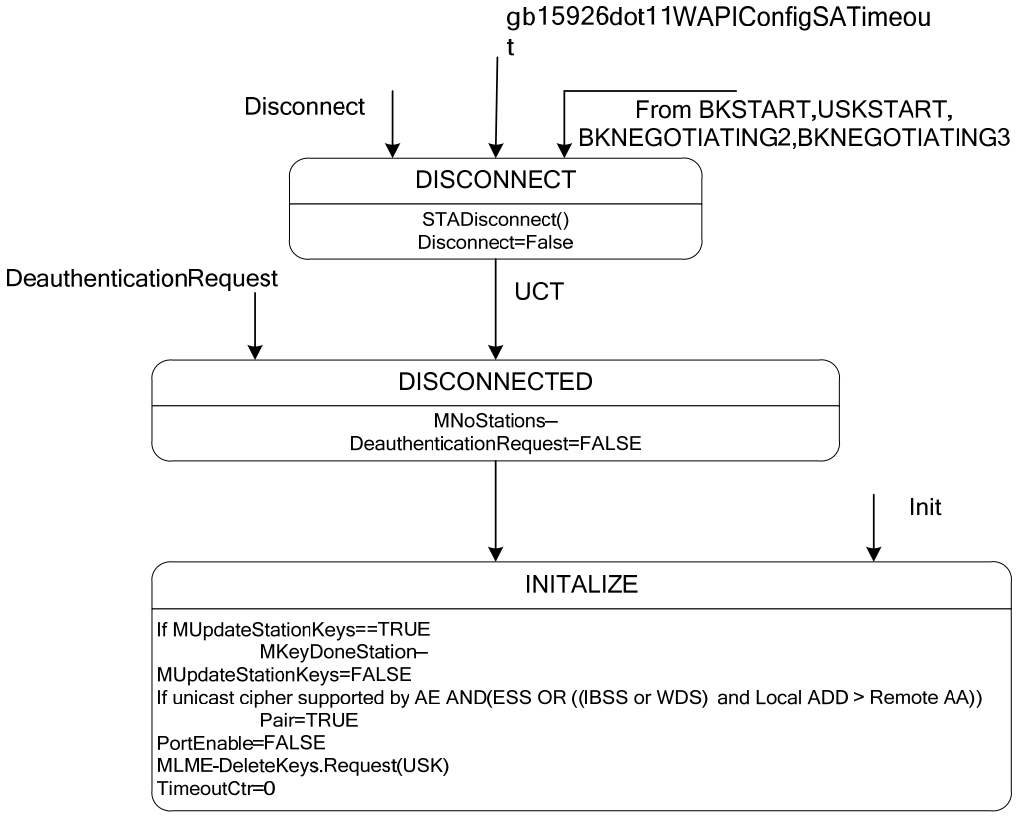


Figure 55 — AE state machine, Part 3

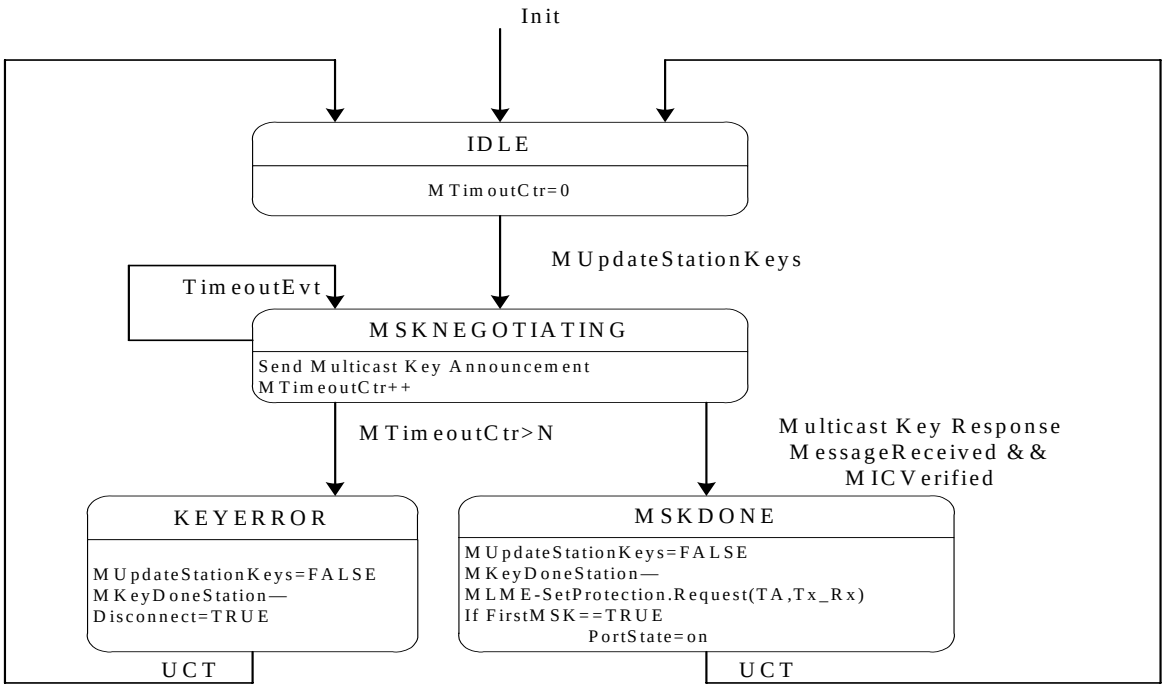


Figure 56 — AE state machine, Part 4

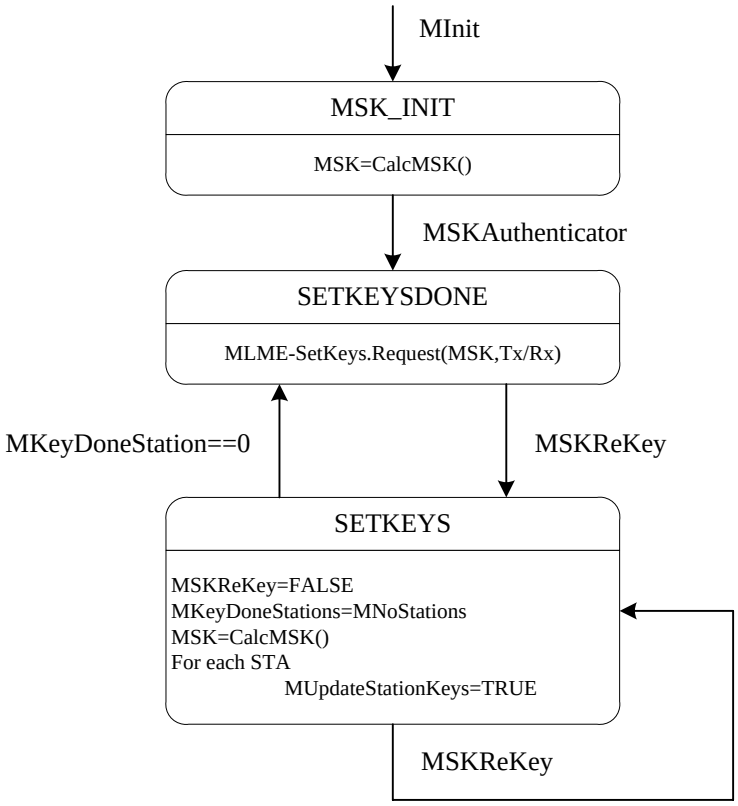


Figure 57 — AE state machine, Part 5

8.3.2.1 AE state machine states

8.3.2.1.1 AE state machine: Certificate Authentication and USK Negotiation (per STA)

The following list summarizes the states which the AE state machine uses to support the Certificate Authentication and the USK Negotiation.

a) AUTHENTICATION: This state is entered when an AuthenticationRequest is sent from the management entity to authenticate a BSSID.

b) BKSTART: This state is entered from AUTHENTICATION to start the BK Negotiation or if the BKReKey is set TRUE.

c) INITPSK: This state is entered when a PSK is configured.

d) BKNEGOTIATING: This state is entered when the Access WAI Authentication Request message is received.

e) BKNEGOTIATING2: This state is entered when the certificate is required to be verified.

f) BKNEGOTIATING3: This state is entered when the Certificate Authentication Response message is received.

g) BKDONE: This state is entered when the certificate is valid and the signature for Access WAI Authentication Request message is verified.

h) USKSTART: This state is entered from BKDONE to start the Unicast Key Negotiation when BKAvailable is set TRUE or if the USKReKey is set TRUE.

i) USKNEGOTIATING: This state is entered when the Unicast Key Negotiation Response message is received.

j) USKDONE: This state is entered when the HMAC for the Unicast Key Negotiation Response message is verified.

k) DISCONNECT: This state is entered if Disconnect is set TRUE or WAI procedure is timed out. It sends a Deauthentication message to the STA and enters the INITIALIZE state.

l) DISCONNECTED: This state is entered when Disassociation or Deauthentication messages are received.

m) INITIALIZE: This state is entered from the DISCONNECTED state, when a Deauthentication request event occurs, or when the station initializes. The state initializes the key state variables.

8.3.2.1.2 AE state machine: Multicast Key Announcement (per STA)

The following list summarizes the states which the AE state machine uses to support the Multicast Key Announcement.

a) IDLE: This state is entered when no Multicast Key Announcement is occurring.

b) KEYERROR: This state is entered if the Multicast Key Response message for the Multicast Key Announcement is not received.

c) MSKDONE: This state is entered when the Multicast Key Response message for the Multicast Key Announcement is received from the ASUE.

d) MSKNEGOTIATING: This state is entered when the MSK is to be sent to the ASUE.

8.3.2.1.3 AE state machine: Multicast Key Announcement (global)

The following list summarizes the states which the AE state machine uses to coordinate a MSK update of all STAs.

a) MSK_INIT: This state is entered on system initialization.

b) SETKEYS: This state is entered if the MSK is to be updated on all ASUEs.

c) SETKEYSDONE: This state is entered if the MSK has been updated on all ASUEs.

8.3.2.2 AE state machine variables

The following list summarizes the variables used by the AE state machine.

a) AuthenticationRequest: This variable is set to TRUE if the STA's ISO/IEC 8802.11 management entity requires an association to be authenticated. This can be set when the STA associates or at other time.

b) DeauthenticationRequest: This variable is set to TRUE if a Disassociation or Deauthentication message is received.

c) Disconnect: This variable is set to TRUE when the STA should initiate a deauthentication.

d) MessageReceived: This variable is set to TRUE when a WAI protocol message is received.

e) MTimeoutCtr: This variable maintains the count of a specific WAI protocol message receive timeouts for the Multicast Key Announcement. It is incremented each time a timeout occurs on a specific WAI protocol message receive event and is initialized to 0.

f) MInit: This variable is used to initialize the MSK state machine.

g) Init: This variable is used to initialize per-STA state machine.

h) TimeoutEvt: This variable is set to TRUE if the WAI protocol message sent out fails to obtain a response from the ASUE. The variable may be set by management action or set by the operation of a timeout while in the BKSTART, USKSTART, BKNEGOTIATING2 and MSKNEGOTIATING states.

i) TimeoutCtr: This variable maintains the count of WAI protocol message receive timeouts. It is incremented each time a timeout occurs on a WAI protocol message receive event and is initialized to 0.

j) HMACVerified: This variable is set to TRUE if the MIC on the received WAI protocol message is verified and is correct. Any WAI protocol message with an invalid HMAC will be dropped and ignored.

k) MSKAE: This variable is set to TRUE if the AE is on an AP or is the designated AE for an IBSS.

l) MKeyDoneStations: Count of number of STAs remaining to have their MSK updated. This is a global variable.

m) MSKRekey: This variable is set to TRUE when a Multicast Key Announcement is required. This is a global variable.

n) BKReKey: This variable is set to TRUE when a BK rekeying is required.

o) USKReKey: This variable is set to TRUE when a USK rekeying is required.

p) MUpdateStationKeys: This variable is set to TRUE when a new MSK is available to be sent to ASUEs.

q) MNoStations: This variable counts the number of AEs so it is known how many ASUEs need to be sent the MSK. This is a global variable.

r) USK: This variable is the current USK.

s) MSK: This variable is the current MSK.

t) BK: This variable is the buffer holding the current BK.

u) Certificate_Verificaton: This variable is set to TRUE when the certificate is required to be verified.

v) PortControlled: This variable is the method by which the port is controlled. The value is Auto or ForceUnauthenticated.

w) Portenabled: This variable is set to TRUE if the port control mechanism is applied.

x) PortState: This variable is set to "on" if the data frame is permitted to pass through the port.

y) Certificate Authentication: This variable is set to TRUE if the WAI certificate authentication and key management are chosen.

z) PSK Authentication: This variable is set to TRUE if the PSK authentication and key management are chosen.

8.3.2.3 AE state machine procedures

The following list summarizes the procedures used by the AE state machine.

a) STADisconnect (): Execution of this procedure deauthenticates the STA.

b) CalcMSK (): Generates the MSK.

9 MAC sublayer functional description

See Clause 9 of ISO/IEC 8802-11:2005, Amd 4:2006 and Amd 5:2006.

10 Layer management

10.1 Overview of management model

See 10.1 of ISO/IEC 8802-11:2005.

10.2 Generic management primitives

See 10.2 of ISO/IEC 8802-11:2005.

10.3 MLME SAP interface

10.3.1 Power management

See 10.3.1 of ISO/IEC 8802-11:2005.

10.3.2 Scan

10.3.2.1 MLME-SCAN.request

See 10.3.2.1 of ISO/IEC 8802-11:2005.

10.3.2.2 MLME-SCAN.confirm

10.3.2.2.1 Function

See 10.3.2.2.1 of ISO/IEC 8802-11:2005.

10.3.2.2.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 13):

MLME-SCAN.confirm (

BSSDescriptionSet,

ResultCode

)

Table 13 — Parameters for MLME-SCAN.confirm

Name	Type	Valid range	Description
BSSDescriptionSet	Set of BSSDescriptions	N/A	The BSSDescriptionSet is returned to indicate the results of the scan request. It is a set containing zero or more instances of a BSSDescription.
ResultCode	Enumeration	SUCCESS, INVALID_PARAMETERS	Indicates the result of the MLME-SCAN.confirm.

Each BSSDescription consists of the following elements (see Table 14).

Table 14 — Elements of BSSDescription

Name	Type	Valid range	Description
BSSID	MACAddress	N/A	The BSSID of the found BSS.
SSID	Octet string	1–32 octets	The SSID of the found BSS.

BSSType	Enumeration	INFRASTRUCTURE, INDEPENDENT	The type of the found BSS.
Beacon Period	Integer	N/A	The Beacon period of the found BSS (in TU).
DTIM Period	Integer	As defined in frame format	The DTIM period of the BSS (in beacon periods).
Timestamp	Integer	N/A	The timestamp of the received frame (probe response/beacon) from the found BSS.
Local Time	Integer	N/A	The value of the STA's TSF timer at the start of reception of the first octet of the timestamp field of the received frame (probe response or beacon) from the found BSS.
PHY parameter set	As defined in frame format	As defined in frame format	The parameter set relevant to the PHY.
CF parameter set	As defined in frame format	As defined in frame format	The parameter set for the CF periods, if found BSS supports CF mode.
IBSS parameter set	As defined in frame format	As defined in frame format	The parameter set for the IBSS, if found BSS is an IBSS.
CapabilityInformation	As defined in frame format	As defined in frame format	The advertised capabilities of the BSS.
BSSBasicRateSet	Set of integers	2–127 inclusive (for each integer in the set)	The set of data rates (in units of 500 kb/s) that must be supported by all STAs that desire to join this BSS. The STAs shall be able to receive at each of the data rates listed in the set.
RSN	RSN information element	As defined in frame format	A description of the cipher suites and AKM suites supported in the BSS.
WAPI	WAPI Parameter Set information element	As defined in frame format	A description of the cipher suites and AKM suites supported in the BSS.
Country	As defined in the Country element	As defined in the Country element	The information required to identify the regulatory domain in which the STA is located and to configure its physical layer (PHY) for operation in that regulatory domain. Present only when TPC functionality is required, as specified in 11.5, or when gb15629dot11MultiDomainCapabilityEnabled is true.
IBSS DFS Recovery Interval	Integer	1–255	Only present if BSSType = INDEPENDENT. The time interval that shall be used for DFS recovery. Present only when DFS functionality is required, as specified in 11.6.

10.3.2.2.3 When generated

See 10.3.2.2.3 of ISO/IEC 8802-11:2005.

10.3.2.2.4 Effect of receipt

See 10.3.2.2.4 of ISO/IEC 8802-11:2005.

10.3.3 Synchronization

See 10.3.3 of ISO/IEC 8802-11:2005.

10.3.4 Open System Authenticate

This mechanism supports the process of establishing an Open System authentication relationship with a peer MAC entity.

10.3.4.1 MLME-AUTHENTICATE .request

10.3.4.1.1 Function

This primitive requests an Open System authentication with a specified peer MAC entity.

10.3.4.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 15):

MLME-AUTHENTICATE.request (

PeerSTAAddress,

OpenSystemAuthenticateFailureTimeout

)

Table 15 — Parameters for MLME-AUTHENTICATE.request

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid Individual MAC address	Specifies the address of the peer MAC entity with which to perform the Open System Authentication process.
OpenSystemAuthentic-ationFailure-Timeout	Integer	≥1	Specifies a time limit (in TU) after which the Open System authentication procedure will be terminated.

10.3.4.1.3 When generated

This primitive is generated by the SME for a STA to establish an Open System authentication with a specified peer MAC entity in order to permit Class 2 frames to be exchanged between the two STAs. During the Open System authentication procedure, the SME may generate additional

MLME-AUTHENTICATE.request primitives.

10.3.4.1.4 Effect of request

This primitive initiates an Open System authentication procedure. The MLME subsequently issues an MLME-AUTHENTICATE.confirm that reflects the results.

10.3.4.2 MLME-AUTHENTICATE.confirm

10.3.4.2.1 Function

This primitive reports the results of an Open System authentication attempt with a specified peer MAC entity.

10.3.4.2.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 16):

MLME-AUTHENTICATE.confirm (

PeerSTAAddress,

ResultCode

)

Table 16 — Parameters for MLME-AUTHENTICATE.confirm

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which the Open System authentication process was attempted. This value must match the peerSTAAddress parameter specified in the corresponding MLME-AUTHENTICATE.request.
ResultCode	Enumeration	SUCCESS INVALID_PARAMETERS TIMEOUT TOO_MANY_ SIMULTANEOUS_ REQUESTS REFUSED	Indicates the result of the MLME-AUTHENTICATE.request.

10.3.4.2.3 When generated

This primitive is generated by the MLME as a result of an MLME-AUTHENTICATE.request to perform the Open System authentication with a specified peer MAC entity.

10.3.4.2.4 Effect of receipt

The SME is notified of the results of the Open System authentication procedure.

10.3.4.3 MLME-AUTHENTICATE.indication

10.3.4.3.1 Function

This primitive reports the establishment of an Open System authentication relationship with a specified peer MAC entity.

10.3.4.3.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 17):

MLME-AUTHENTICATE.indication (

PeerSTAAddress

)

Table 17 — Parameters for MLME-AUTHENTICATE.indication

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC address with which the Open System authentication relationship was established.

10.3.4.3.3 When generated

This primitive is generated by MLME as a result of the establishment of an Open System authentication relationship with a specified peer MAC entity that resulted from an Open System authentication procedure that is initiated by that specified peer MAC entity.

10.3.4.3.4 Effect of receipt

The SME is notified of the establishment of the Open System authentication relationship.

10.3.5 DEAUTHENTICATE

This mechanism supports the process of invalidating an Open System authentication relationship with a peer MAC entity.

10.3.5.1 MLME-DEAUTHENTICATE.request

10.3.5.1.1 Function

This primitive requests that the Open System authentication relationship with a specified peer MAC entity be invalidated.

10.3.5.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 18):

MLME-DEAUTHENTICATE.request (

PeerSTAAddress

ReasonCode

)

Table 18 — Parameters for MLME-DEAUTHENTICATE.request

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which to invalidate the relationship of the Open System authentication.
ReasonCode	As defined in frame format	As defined in frame format	Specifies the reason to invalidate the Open System authentication.

10.3.5.1.3 When generated

This primitive is generated by the SME for a STA to invalidate the Open System authentication with a specified peer MAC entity in order to prevent the exchange of Class 2 frames between the two STAs. During the invalidation procedure, the SME may generate additional MLME-DEAUTHENTICATE.request primitives.

10.3.5.1.4 Effect of receipt

This primitive initiates the invalidation procedure of the Open System authentication. The MLME subsequently issues an MLME-DEAUTHENTICATE .confirm that reflects the results.

10.3.5.2 MLME-DEAUTHENTICATE.confirm

10.3.5.2.1 Function

This primitive reports the result of a deauthentication attempt with a specified peer MAC entity.

10.3.5.2.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 19):

MLME-DEAUTHENTICATE.confirm (

PeerSTAAddress,

ResultCode

)

Table 19 — Parameters for MLME-DEAUTHENTICATE.confirm

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which the invalidation process of the Open System authentication was attempted.

ResultCode	Enumeration	SUCCESS INVALID_PARAMETERS TOO_MANY_ SIMULTANEOUS_ REQUESTS	Indicates the result of the MLME-DEAUTHENTICATE.request.

10.3.5.2.3 When generated

This primitive is generated by the MLME as the result of an MLME-DEAUTHENTICATE.request to invalidate the Open System authentication relationship with a specified peer MAC entity.

10.3.5.2.4 Effect of receipt

The SME is notified of the results of the Open System deauthentication procedure.

10.3.5.3 MLME-DEAUTHENTICATE.indication

10.3.5.3.1 Function

This primitive reports the invalidation of an Open System authentication relationship with a specified peer MAC entity.

10.3.5.3.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 20):

MLME-DEAUTHENTICATE.indication (

PeerSTAAddress,

ReasonCode

)

Table 20 — Parameters for MLME-DEAUTHENTICATE.indication

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC address with which the Open System authentication relationship is invalidated.
ReasonCode	As defined in frame format	As defined in frame format	Specifies the reason the deauthentication procedure is initiated.

10.3.5.3.3 When generated

This primitive is generated by the MLME as a result of the invalidation of the Open System authentication relationship with specified peer MAC entity.

10.3.5.3.4 Effect of receipt

The SME is notified of the invalidation of the specific Open System authentication relationship.

10.3.6 Associate

10.3.6.1 MLME-ASSOCIATE.request

10.3.6.1.1 Function

See 10.3.6.1.1 of ISO/IEC 8802-11:2005.

10.3.6.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 21):

MLME-ASSOCIATE.request(

PeerSTAAddress,

AssociateFailureTimeout,

CapabilityInformation,

ListenInterval,

Supported Channels,

RSN,

WAPI

)

Table 21 — Parameters for MLME-ASSOCIATE.request

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which to perform the association process
AssociateFailureTimeout	Integer	≥1ı	Specifies a time limit (in TU) after which the associate procedure will be terminated
CapabilityInformation	As defined inframe format	As defined inframe format	Specifies the operational capability definitions to be used by the MAC entity
ListenInterval	Integer	≥0	Specifies the number of beacon intervals that may pass before the STA awakens and listens for the next beacon
Supported Channels	As defined in the Supported Channels element	As defined in the Supported Channels element	The list of channels in which the STA is capable of operating. Present only when DFS functionality is required, as

			specified in 11.6.
RSN	RSN information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports
WAPI	WAPI Parameter Set information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports

10.3.6.1.3 When generated

See 10.3.6.1.3 of ISO/IEC 8802-11:2005.

10.3.6.1.4 Effect of receipt

See 10.3.6.1.4 of ISO/IEC 8802-11:2005.

10.3.6.2 MLME-ASSOCIATE.confirm

See 10.3.6.2 of ISO/IEC 8802-11:2005.

10.3.6.3 MLME-ASSOCIATE.indication

10.3.6.3.1 Function

See 10.3.6.3.1 of ISO/IEC 8802-11:2005.

10.3.6.3.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 22):

MLME-ASSOCIATE.indication(

PeerSTAAddress,

RSN,

WAPI

)

Table 22 — Parameters for MLME-ASSOCIATE.indication

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which the association was established
RSN	RSN information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports

WAPI	WAPI Parameter Set information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports. There is only one unicast cipher suite in the WAPI Parameter Set information element

10.3.6.3.3 When generated

See 10.3.6.3.3 of ISO/IEC 8802-11:2005.

10.3.6.3.4 Effect of receipt

See 10.3.6.3.4 of ISO/IEC 8802-11:2005.

10.3.7 Reassociate

10.3.7.1 MLME-REASSOCIATE.request

10.3.7.1.1 Function

See 10.3.7.1.1 of ISO/IEC 8802-11:2005.

10.3.7.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 23):

MLME-REASSOCIATE.request(

NewAPAddress,

ReassociateFailureTimeout,

CapabilityInformation,

ListenInterval,

Supported Channels,

RSN,

WAPI

)

Table 23 — Parameters for MLME-REASSOCIATE.request

Name	Type	Valid range	Description
NewAPAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which to perform the reassociation process
ReassociateFailureTimeout	Integer	≥ 1	Specifies a time limit (in TU) after which the reassociate procedure will be terminated
CapabilityInformation	As defined in frame format	As defined in frame format	Specifies the operational capability definitions to be used by the MAC entity
ListenInterval	Integer	≥ 0	Specifies the number of beacon intervals that may pass before the STA awakens and listens for the next beacon.
Supported Channels	As defined in the Supported Channels element	As defined in the Supported Channels element	The list of channels in which the STA is capable of operating. Present only when DFS functionality is required, as specified in 11.6.
RSN	RSN information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports
WAPI	WAPI Parameter Set information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports

10.3.7.1.3 When generated

See 10.3.7.1.3 of ISO/IEC 8802-11:2005.

10.3.7.1.4 Effect of receipt

See 10.3.7.1.4 of ISO/IEC 8802-11:2005.

10.3.7.2 MLME-REASSOCIATE.confirm

See 10.3.7.2 of ISO/IEC 8802-11:2005.

10.3.7.3 MLME-REASSOCIATE.indication

10.3.7.3.1 Function

See 10.3.7.3.1 of ISO/IEC 8802-11:2005.

10.3.7.3.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 24):

MLME-REASSOCIATE.indication(

PeerSTAAddress,

RSN,

WAPI

)

Table 24 — Parameters for MLME-REASSOCIATE.indication

Name	Type	Valid range	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address	Specifies the address of the peer MAC entity with which the association was established
RSN	RSN information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports
WAPI	WAPI Parameter Set information element	As defined in frame format	Specifies the cipher suites and AKM suites that the BSS supports. There is only one unicast cipher suite in the WAPI Parameter Set information element.

10.3.7.3.3 When generated

See 10.3.7.3.3 of ISO/IEC 8802-11:2005.

10.3.7.3.4 Effect of receipt

See 10.3.7.3.4 of ISO/IEC 8802-11:2005.

10.3.8 Disassociate

See 10.3.8 of ISO/IEC 8802-11:2005.

10.3.9 Reset

See 10.3.9 of ISO/IEC 8802-11:2005.

10.3.10 Start

See 10.3.10 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.11 Spectrum management protocol layer model

See 10.3.11 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.12 Measurement request

See 10.3.12 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.13 Channel measurement

See 10.3.13 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.14 Measurement report

See 10.3.14 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.15 Channel switch

See 10.3.15 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.16 10.3.16 TPC request

See 10.3.16 of ISO/IEC 8802-11:2005 and Amd 5:2006.

10.3.17 SETWPIKEYS

10.3.17.1 MLME-SETWPIKEYS.request

10.3.17.1.1 Function

This primitive causes an initiator MAC entity to set the appropriate keys with specified peer MAC entity when WPI is enabled.

10.3.17.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 25):

MLME-SETWPIKEYS.request(

Keylist

)

Table 25 — Parameters for MLME-SETWPIKEYS.request

Name	Type	Description
Keylist	A set of SetKeyDescriptors	The list of keys to be used by the MAC.

Each SetKeyDescriptor consists of the following elements (see Table 26):

Table 26 — Elements of the SetKeyDescriptor

Name	Type	Description
Key	Octet string	Encryption key and integrity check key values.
Length	Integer	The number of octets in the key.
KeyIdx	Integer	The number of the key, with a value of 0 or 1.
KeyType	Integer	Defines whether this key is a MSK, USK or STAKey.
PeerSTAAddress	MACAddress	Any valid individual MAC address. This parameter is valid only when the KeyType value is Unicast, or when the KeyType value is Multicast and the STA is in an IBSS, or when the KeyType value is STAKey.
AE/ASUE or Initiator/Peer	Boolean	Whether the key is configured by the AE (Initiator) or ASUE (Peer); true indicates AE or Initiator; false indicates ASUE or Peer.
GSN	Integer	The sequence number of the currently encrypted multicast message. It is valid only when the KeyType value is MSK.
Cipher Suite Selector	4 octets	As defined in the WAPI Parameter Set information element format, it is the cipher suite required for this association.

10.3.17.1.3 When generated

This primitive is generated by the SME at any time when the key negotiation is completed or the preshared key is set.

10.3.17.1.4 Effect of receipt

This primitive enables the MAC to perform the privacy of the data using the appropriate keys. The MLME subsequently issues an MLME-SETWPIKEYS.confirm that reflects the results.

10.3.17.2 MLME-SETWPIKEYS.confirm

10.3.17.2.1 Function

This primitive confirms that the action of the associated MLME-SETWPIKEYS.request primitive has been completed.

10.3.17.2.2 Semantics of the service primitive

This primitive has no parameters.

10.3.17.2.3 When generated

This primitive is generated by the MAC in response to receipt of an MLME-SETWPIKEYS.request primitive. This primitive is issued when the action requested has been completed.

10.3.17.2.4 Effect of receipt

The SME is notified that the requested action of the MLME-SETWPIKEYS.request primitive is completed.

10.3.18 DELETEWPIKEYS

10.3.18.1 MLME-DELETEWPIKEYS.request

10.3.18.1.1 Function

This primitive causes an initiator MAC entity to delete the appropriate keys with specified peer MAC entity when WPI is enabled.

10.3.18.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 27):

MLME-DELETEWPIKEYS.request(

Keylist

)

Table 27 — Parameters of MLME-DELETEWPIKEYS.request

Name	Type	Description
Keylist	A set of DeleteKeyDescriptors	The list of keys to be deleted from the MAC.

Each DeleteKeyDescriptor consists of the following elements (see Table 28):

Table 28 — Elements of the DeleteKeyDescriptor

Name	Type	Description
PeerSTAAddress	MACAddress	Any valid individual MAC address. This parameter is valid only when the KeyType value is Unicast, or when the KeyType value is Multicast and is from an IBSS STA, or when the KeyType value is STAKey.
KeyIdx	Integer	The number of the key, with a value of 0 or 1.
KeyType	Integer	Defines whether this key is a MSK, USK or STAKey.

10.3.18.1.3 When generated

This primitive is generated by the SME at any time when a STA wants to delete the appropriate keys.

10.3.18.1.4 Effect of receipt

This primitive disables the MAC to perform the confidentiality of the data using the appropriate keys. The MLME subsequently issues an MLME- DELETEWPIKEYS.confirm that reflects the results.

10.3.18.2 MLME-DELETEWPIKEYS.confirm

10.3.18.2.1 Function

This primitive confirms that the action of the associated MLME-DELETEWPIKEYS.request primitive has been completed.

10.3.18.2.2 Semantics of the service primitive

This primitive has no parameters.

10.3.18.2.3 When generated

This primitive is generated by the MLME MAC in response to receipt of an MLME-DELETEWPIKEYS.request primitive. This primitive is issued when the action requested has been completed.

10.3.18.2.4 Effect of receipt

The SME is notified that the requested action of the MLME-DELETEWPIKEYS.request primitive is completed.

10.3.19 MLME-STAKEYESTABLISHED

See 10.3.21 of ISO/IEC 8802-11:2005.

10.3.20 SetProtection

10.3.20.1 MLME-SETPROTECTION.request

10.3.20.1.1 Function

See 10.3.22.1.1 of ISO/IEC 8802-11:2005.

10.3.20.1.2 Semantics of the service primitive

The primitive parameters are as follows (see Table 29):

MLME-SETPROTECTION.request(

Protectlist

)

Table 29 — Parameters for MLME-SETPROTECTION.request

Name	Type	Description
Protectlist	A set of protection elements	The list of how each key is being used currently.

Each Protectlist consists of the following elements (see Table 30):

Table 30 — Elements of the Protectlist

Name	Type	Description
Address	MACAddress	Any valid individual MAC address. This parameter is valid only when the KeyType value is Unicast or STAKey or when the KeyType value is Multicast and is from an IBSS STA.
ProtectType	Enumeration (None, Rx, Tx, Rx_Tx)	The protection value for this MAC.
KeyType	Integer	Defines whether this key is a MSK, USK or STAKey.

10.3.20.1.3 When generated

See 10.3.22.1.3 of ISO/IEC 8802-11:2005.

10.3.20.1.4 Effect of receipt

See 10.3.22.1.4 of ISO/IEC 8802-11:2005.

10.3.20.2 MLME-SETPROTECTION.confirm

See 10.3.22.2 of ISO/IEC 8802-11:2005.

10.3.21 MLME-PROTECTEDFRAMEDROPPED

See 10.3.23 of ISO/IEC 8802-11:2005.

10.4 PLME SAP interface

See 10.4 of ISO/IEC 8802-11:2005 and Amd 4:2006.

11 MAC sublayer management entity

11.1 Synchronization

See 11.1 of ISO/IEC 8802-11:2005.

11.2 Power management

See 11.2 of ISO/IEC 8802-11:2005.

11.3 Association and reassociation

This subclause describes the procedures used in accordance with ISO/IEC 8802-11 Open System authentication and deauthentication. The states used in this description are those defined in 5.5.

11.3.1 Authentication — Originating STA

Upon receipt of an MLME-AUTHENTICATE.request primitive, the originating STA shall authenticate with the indicated STA using the following procedure.

a) In an ESS, or optionally in an IBSS, the STA shall execute the Open System authentication mechanism.

b) If the Open System authentication is successful, the state variable of the indicated STA shall be set to State 2.

c) The STA shall issue an MLME-AUTHENTICATE.confirm primitive to inform the SME of the result of the Open System authentication.

The STA's SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using MLME-DELETEWPIKEYS.request primitive before invoking the MLME-AUTHENTICATE.request primitive.

11.3.2 Authentication — Destination STA

Upon receipt of an Open System authentication frame with authentication transaction sequence number equal to 1, the destination STA shall authenticate with the indicated STA using the following procedure.

a) The STA shall execute the Open System authentication mechanism.

b) The STA shall issue an MLME-AUTHENTICATE.indication primitive to inform the SME of the Open System authentication.

The STA's SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using the MLME-DELETEWPIKEYS.request primitive upon receiving an MLME-AUTHENTICATE.indication primitive.

If the STA is in an IBSS, and if the SME decides to initiate a WAPI association, and if the SME does not know the security policy of the peer, it may issue a unicast Probe Request frame to the peer by invoking an MLME-SCAN.request to discover the peer's security policy.

11.3.3 Deauthentication — Originating STA

See 11.3.3 of ISO/IEC 8802-11:2005.

The STA shall issue an MLME-DEAUTHENTICATE.confirm primitive to inform the SME of the completion of the Deauthentication.

11.3.4 Deauthentication — Destination STA

See 11.3.4 of ISO/IEC 8802-11:2005.

11.4 Association, reassociation, and disassociation

11.4.1 STA association procedures

Upon receipt of an MLME-ASSOCIATE.request primitive, a STA shall associate with an AP via the following procedure.

a) The STA shall transmit an Association Request frame to the AP with which that STA is authenticated. If the MLME-ASSOCIATE.request primitive contains a WAPI Parameter Set information element with only one unicast cipher suite and only one authenticated key suite, this WAPI Parameter Set information element shall be included in the Association Request frame.

b) If an Association Response frame is received with a status value of "successful," the STA is now associated with the AP. The state variable shall be set to State 3, and the MLME shall issue an MLME-ASSOCIATE.confirm primitive indicating the successful completion of the operation.

c) If an Association Response frame is received with a status value other than “successful” or the AssociateFailureTimeout expires, the STA is not associated with the AP. The MLME shall issue an MLME-ASSOCIATE.confirm primitive indicating the failure of the operation.

d) The SME shall establish a WAPI association by calling MLME.SETPROTECTION.request primitive with ProtectType set to “Rx_Tx,” or it shall do nothing if it does not wish to secure communication.

The STA’s SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using MLME-DELETEWPIKEYS.request primitive before invoking MLME-ASSOCIATE.request primitive.

11.4.2 AP association procedures

When an Association Request frame is received from a STA, the AP shall associate with the STA using the following procedure.

a) If the STA is not authenticated, the AP shall transmit a Deauthentication frame to the STA and terminate the association procedure.

b) In a WAPI association, the AP shall check if the values received in the WAPI Parameter Set information element match the AP’s security policy. If not, the association shall not be accepted.

c) The AP shall transmit an Association Response. If the status value is “successful,” the association identifier assigned to the STA shall be included in the response.

d) When the Association Response with a status value of “successful” is acknowledged by the STA, the STA is considered to be associated with this AP. The state variable for the STA shall be set to State 3.

e) The MLME shall issue an MLME-ASSOCIATE.indication primitive to inform the SME of the association.

f) The SME shall establish a WAPI association by calling MLME.SETPROTECTION.request primitive with ProtectType set to “Rx_Tx,” or it shall do nothing if it does not wish to secure communication.

g) The SME will inform the DS of the new association.

The STA’s SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using MLME-DELETEWPIKEYS.request primitive upon receiving an MLME-ASSOCIATE.indication primitive.

11.4.3 STA reassociation procedures

Upon receipt of an MLME-REASSOCIATE.request primitive, a STA shall reassociate with an AP via the following procedure.

a) If the state variable is in State 1, the STA shall inform the SME of the failure of the reassociation by issuing an MLME-REASSOCIATE.confirm primitive.

b) The STA shall transmit a Reassociation Request frame to the new AP. If the MLME-REASSOCIATE.request primitive contains a WAPI Parameter Set information element with only one unicast cipher suite and only one authenticated key suite, this WAPI Parameter Set information element shall be included in the Reassociation Request frame.

c) If a Reassociation Response frame is received with a status value of “successful,” the STA is now associated with the new AP. The state variable shall be set to State 3, and the MLME shall issue an MLME-REASSOCIATE.confirm primitive indicating the successful completion of the operation.

d) If a Reassociation Response frame is received with a status value other than “successful” or the AssociateFailureTimeout expires, the STA is not associated with the AP. The MLME shall issue an MLME-REASSOCIATE.confirm primitive indicating the failure of the operation.

e) The SME shall establish a WAPI association calling MLME.SETPROTECTION.request primitive with ProtectType set to “Rx_Tx,” or it shall do nothing if it does not wish to secure communication.

The STA’s SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using MLME-DELETEWPIKEYS.request primitive before invoking MLME-REASSOCIATE.request primitive.

11.4.4 AP reassociation procedures

Whenever a Reassociation Request frame is received from a STA, the AP uses the following procedure to support reassociation.

a) If the STA is not authenticated, the AP shall transmit a Deauthentication frame to the STA and terminate the reassociation procedure.

b) In a WAPI association, the AP shall check if the values received in the WAPI Parameter Set information element match the AP’s security policy. If not, the association shall not be accepted.

c) The AP shall transmit a Reassociation Response frame. If the status value is “successful,” the association identifier assigned to the STA shall be included in the response.

d) When the Reassociation Response frame with a status value of “successful” is acknowledged by the STA, the STA is considered to be associated with this AP. The state variable of the STA shall be set to State 3.

e) The MLME shall issue an MLME-REASSOCIATE.indication primitive to inform the SME of the association.

f) The SME shall establish a WAPI association called MLME.SETPROTECTION.request primitive with ProtectType set to “Rx_Tx,” or it shall do nothing if it does not wish to secure communication.

g) The SME will inform the DS of the new association.

The STA’s SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using MLME-DELETEWPIKEYS.request primitive upon receiving an MLME-REASSOCIATE.indication primitive.

11.4.5 STA disassociation procedures

Upon receipt of an MLME-DISASSOCIATE.request primitive, an associated STA shall disassociate from an AP using the following procedure.

a) The STA shall transmit a Disassociation frame to the AP with which that STA is associated.

b) The state variable of the AP shall be set to State 2 if and only if it was not State 1.

c) The MLME shall issue an MLME-DISASSOCIATE.confirm primitive indicating the successful completion of the operation.

The STA’s SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using the MLME-DELETEWPIKEYS.request primitive and by invoking MLME-SETPROTECTION.request (None) before invoking an MLME-DISASSOCIATE.request primitive.

11.4.6 AP disassociation procedures

Upon receipt of a Disassociation frame from an associated STA, the AP shall disassociate the STA via the following procedure.

a) The state variable of the STA shall be set to State 2.

b) The MLME shall issue an MLME-DISASSOCIATE.indication primitive to inform the SME of the disassociation.

c) The SME will update the DS.

The STA's SME shall delete any USKSA and temporal keys held for communication with the indicated STA by using an MLME-DELETEWPIKEYS.request primitive and by invoking MLME-SETPROTECTION.request (None) upon receiving an MLME-DISASSOCIATE.indication primitive.

11.5 TPC procedures

See 11.1 of ISO/IEC 8802-11:2005/Amd 5:2006.

11.6 DFS procedures

See 11.1 of ISO/IEC 8802-11:2005/Amd 5:2006.

12 PHY service specification

See Clause 12 of ISO/IEC 8802-11:2005.

13 PHY management

See Clause 13 of ISO/IEC 8802-11:2005.

14 Frequency-Hopping spread spectrum (FHSS) PHY specification for the 2.4 GHz industrial, scientific and medical (ISM) band

See Clause 14 of ISO/IEC 8802-11:2005.

15 DSSS PHY specification for the 2.4 GHz band designated for ISM applications

See Clause 15 of ISO/IEC 8802-11:2005.

16 Infrared (IR) PHY specification

See Clause 16 of ISO/IEC 8802-11:2005.

17 Orthogonal frequency division multiplexing (OFDM) PHY specification for the 5 GHz band

See Clause 17 of ISO/IEC 8802-11:2005 and Amd 5:2006.

18 High Rate direct sequence spread spectrum (HR/DSSS) PHY specification

See Clause 18 of ISO/IEC 8802-11:2005 and Amd 4:2006.

19 Extended Rate PHY specification

See Clause 19 of ISO/IEC 8802-11:2005/Amd 4:2006.

Annex A
(normative)

Protocol Implementation Conformance Statements (PICS)

A.1 Introduction

See A.1 of ISO/IEC 8802-11:2005.

A.2 Abbreviations and special symbols

See A.2 of ISO/IEC 8802-11:2005.

A.3 Instructions for completing the PICS proforma

See A.1 of ISO/IEC 8802-11:2005 and Amd 4:2006.

A.4 PICS proforma

A.4.1 Implementation identification

See A.1 of ISO/IEC 8802-11:2005.

A.4.2 Protocol summary

See A.1 of ISO/IEC 8802-11:2005.

A.4.3 IUT configuration

See A.1 of ISO/IEC 8802-11:2005, Amd 4:2006 and Amd 5:2006.

A.4.4 A.4.4 MAC protocol

A.4.4.1 MAC protocol capabilities

MAC protocol capabilities are described in Table A.1, in which support for the MAC protocol capabilities can be indicated.

Table A.1 — MAC protocol capabilities

Item	Protocol capability	References	Status	Support
PC1	Authentication service	5.4.3.1, 5.4.3.2, 5.7.6, 5.7.7,	M	Yes ❑ No ❑

		Annex C		
PC1.1	Authentication state	5.5	M	Yes ❑ No ❑
PC1.2	Open System Authentication	5.7.6	M	Yes ❑ No ❑
PC2	Authentication and Confidentiality	7.2.2, 7.3.1.4, 5.4.3.3, 8.1, 11.3, 11.4, 8.2	O	Yes ❑ No ❑
PC2.1	WAPI Parameter Set information element (IE)	7.3.2.25	PC2:M	Yes ❑ No ❑
PC2.1.1	Multicast cipher suite	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.2	Unicast cipher suite list	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.2.1	WPI data confidentiality	8.2	PC2.1:M	Yes ❑ No ❑
PC2.1.3	Authentication key management (AKM) suite list	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.3.1	WAI Certificate Authentication and Key Management	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.3.2	WAI Preshared Key Authentication and Key Management	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3	WAI Authentication and key management	8.1	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3.1	Key derivation	8.1.4.10	PC2.1:M	Yes ❑ No ❑
PC34.1.3.3.1.1	Base key derivation	8.1.4.10.1	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3.1.2	Unicast session key derivation	8.1.4.10.2	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3.1.3	Multicast session key derivation	8.1.4.10.3	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3.2	Unicast Key Negotiation	8.1.4.3	PC2.1:M	Yes ❑ No ❑
PC2.1.3.3.3	Multicast Key/STAKey Announcement	8.1.4.4	PC2.1:M	Yes ❑ No ❑
PC2.1.4	WAPI capabilities	7.3.2.25	PC2.1:M	Yes ❑ No ❑
PC2.1.5	WAPI Preauthentication	8.1.4.6	PC2.1:O	Yes ❑ No ❑

PC2.1.6	WAPI security association management	8.1.2	PC2.1:M	Yes ❑ No ❑
PC2.1.7	BKSA caching	8.1.4.7	PC2.1:M	Yes ❑ No ❑
PC2.1.8	WAPI extended service set (ESS)	8.1.2.2.1	(PC2.1 and CF1):M	Yes ❑ No ❑
PC2.1.8.1	WAPI STAKey	8.1.4.5	PC2.1.8:O	Yes ❑ No ❑
PC2.1.9	WAPI independent basic service set (IBSS)	8.1.2.2.2	(PC2.1 and CF2):O	Yes ❑ No ❑
PC2.1.10	Rekeying	8.1.4.8	PC2.1.8:O	Yes ❑ No ❑
PC2.1.11	X.509 v3 certificate	8.1.3.1	PC2.1:M	Yes ❑ No ❑
PC2.1.12	GBW certificate	8.1.3.2	PC2.1:O	Yes ❑ No ❑

A.4.4.2 MAC frames

See A.4.4.2 of ISO/IEC 8802-11:2005.

A.4.4.3 Frame exchange sequences

See A.4.4.3 of ISO/IEC 8802-11:2005.

A.4.4.4 MAC addressing functions

See A.4.4.4 of ISO/IEC 8802-11:2005.

A.4.5 Frequency hopping (FH) PHY functions

See A.4.5 of ISO/IEC 8802-11:2005.

A.4.6 Direct sequence PHY functions

See A.4.6 of ISO/IEC 8802-11:2005.

A.4.7 IR baseband PHY functions

See A.4.7 of ISO/IEC 8802-11:2005.

A.4.8 OFDM PHY functions

See A.4.8 of ISO/IEC 8802-11:2005.

A.4.9 High Rate, direct sequence PHY functions

See A.4.9 of ISO/IEC 8802-11:2005.

A.4.10 Regulatory Domain Extensions

See A.4.10 of ISO/IEC 8802-11:2005.

A.4.11 ERP Physical Layer functions

See A.4.11 of ISO/IEC 8802-11:2005/Amd 4:2005.

A.4.12 Spectrum management extensions

See A.4.12 of ISO/IEC 8802-11:2005/Amd 5:2005.

Annex B
(informative)

Hopping sequences

See Annex B of ISO/IEC 8802-11:2005.

Annex C
(normative)

Formal description of MAC operation

See Annex C of ISO/IEC 8802-11:2005 and Amd 4:2006.

Annex D
(normative)

ASN.1 encoding of the MAC and PHY MIB

```
-- ********************************************

-- * IEEE 802.11 MIB

-- ********************************************
```

See Annex D of ISO/IEC 8802-11:2005, Amd 4:2006, Amd 5:2006 and Amd 6:2006.

```
-- ********************************************

-- *  GB15629dot11-WAPI-MIB

-- ********************************************

GB15629dot11-WAPI-MIB DEFINITIONS ::= BEGIN

   IMPORTS

      MODULE-IDENTITY,OBJECT-TYPE,Counter32,

      Unsigned32                          FROM SNMPv2-SMI

      MacAddress,TruthValue,DisplayString          FROM SNMPv2-TC

      MODULE-COMPLIANCE,OBJECT-GROUP            FROM SNMPv2-CONF

      ifIndex                             FROM RFC1213-MIB ;


-- ************************************************

-- *  MODULE IDENTITY

-- ************************************************

gb15629dot11wapiMIB MODULE-IDENTITY

   ORGANIZATION

      “ChinaBWIPS（China Broadband Wireless IP Standard Group )”

      CONTACT-INFO

      “Address：P.O.BOX 100191.ext.006, Beijing, China.

      Postcode：100191
```

```
        Tel: +86 10 82356842

        Fax: +86 10 82350471

        E-mail: bwips@chinabwips.org”

    DESCRIPTION

            “The MIB module for WAPI entities.”

    ::= { iso(1)  member-body(2)  cn(156)  bwips(11235)  GB15629(15629)  GB15629-11(11)  GB15629-11-
mibs(1) 1 }


-- ************************************

-- *  Major sections

-- ************************************

wapiMIBObjects OBJECT IDENTIFIER              ::= { gb15629dot11wapiMIB 1 }

wapiMIBConformance OBJECT IDENTIFIER          ::= { gb15629dot11wapiMIB 2 }


-- ****************************************

-- *   wapiMIBObjects table

-- ****************************************

-- gb15629dot11wapiConfig                         ::= { wapiMIBObjects 1 }

-- gb15629dot11wapiConfigUnicastCiphers           ::= { wapiMIBObjects 2 }

-- gb15629dot11wapiConfigAuthenticationSuites  ::= { wapiMIBObjects 3 }

-- gb15629dot11wapiStats                          ::= { wapiMIBObjects 4 }


--*************************************************************************

-- *   gb15629dot11wapiConfig table

-- *************************************************************************

        gb15629dot11wapiConfig OBJECT-TYPE

                SYNTAX SEQUENCE OF gb15629dot11wapiConfigEntry

                MAX-ACCESS not-accessible

                STATUS current

                DESCRIPTION
```

"Station Configuration attributes. In tablular form to allow for multiple instances on an agent."

::={ wapiMIBObjects 1 }

gb15629dot11wapiConfigEntry OBJECT-TYPE

SYNTAX gb15629dot11wapiConfigEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry in the gb15629dot11wapiConfigTable. It is possible for there to be multiple WAPI interfaces on one agent, each with its unique MAC address. The relationship between a WAPI interface and an interface in the context of the Internet-standard MIB is one-to-one. As such, the value of an ifIndex object instance can be directly used to identify corresponding instances of the objects defined herein.

ifIndex - Each WAPI interface is represented by an ifEntry. Interface tables in this MIB module are indexed by ifIndex."

INDEX {ifIndex}

::={ gb15629dot11wapiConfig 1 }

gb15629dot11wapiConfigEntry::=

SEQUENCE {

gb15629dot11wapiConfigVersion	Integer32,
gb15629dot11wapiControlledAuthControl	TruthValue,
gb15629dot11wapiControlledPortControl	INTEGER,
gb15629dot11wapiOptionalImplemented	TruthValue,
gb15629dot11wapiPreauthenticationImplemented	TruthValue,
gb15629dot11wapiEnabled	TruthValue,
gb15629dot11wapiPreauthenticationEnabled	TruthValue,
gb15629dot11wapiConfigUnicastKeysSupported	Unsigned32,
gb15629dot11wapiConfigUnicastRekeyMethod	INTEGER,
gb15629dot11wapiConfigUnicastRekeyTime	Unsigned32,
gb15629dot11wapiConfigUnicastRekeyMessages	Unsigned32,
gb15629dot11wapiConfigMulticastCipher	OCTET STRING,

```
        gb15629dot11wapiConfigMulticastRekeyMethod       INTEGER,
        gb15629dot11wapiConfigMulticastRekeyTime         Unsigned32,
        gb15629dot11wapiConfigMulticastRekeyMessages     Unsigned32,
        gb15629dot11wapiConfigMulticastRekeyStrict       TruthValue,
        gb15629dot11wapiConfigPSKValue                   OCTET STRING,
        gb15629dot11wapiConfigPSKPassPhrase              DisplayString,
        gb15629dot11wapiConfigCertificateUpdateCount     Unsigned32,
        gb15629dot11wapiConfigMulticastUpdateCount       Unsigned32,
        gb15629dot11wapiConfigUnicastUpdateCount         Unsigned32,
        gb15629dot11wapiConfigMulticastCipherSize        Unsigned32,
        gb15629dot11wapiConfigBKLifetime                 Unsigned32,
        gb15629dot11wapiConfigBKReauthThreshold          Unsigned32,
        gb15629dot11wapiConfigSATimeout                  Unsigned32,
        gb15629dot11wapiAuthenticationSuiteSelected      OCTET STRING,
        gb15629dot11wapiUnicastCipherSelected            OCTET STRING,
        gb15629dot11wapiMulticastCipherSelected          OCTET STRING,
        gb15629dot11wapiBKIDUsed                         OCTET STRING,
        gb15629dot11wapiAuthenticationSuiteRequested     OCTET STRING,
        gb15629dot11wapiUnicastCipherRequested           OCTET STRING,
        gb15629dot11wapiMulticastCipherRequested         OCTET STRING }

gb15629dot11wapiConfigVersion OBJECT-TYPE
        SYNTAX Integer32
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            “This object indicates the highest version that this entity supports.”
    ::= { gb15629dot11wapiConfigEntry 1}
```

gb15629dot11wapiControlledAuthControl OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This attribute indicates whether the entity enable the authentication. If the value is "0" which means "authentication disabled", the state of the controlled port shall be set to "authenticated"; else the value is "1" which means "authentication enabled", the state of the controlled port shall be based on the gb15629dot11wapiControlledPortControl."

::= { gb15629dot11wapiConfigEntry 2}

gb15629dot11wapiControlledPortControl OBJECT-TYPE

SYNTAX INTEGER

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This attribute, when the gb15629dot11wapiControlledAuthControl is set to true, shall be effective and indicate the methods by which the port is controlled. If the value is "0" which means "auto", the state of the controlled port shall be based on the result of the authentication; else the value is "1" which means "force unauthenticated", the state of the controlled port shall be unconditionally set to "unauthenticated"."

::= { gb15629dot11wapiConfigEntry 3}

gb15629dot11wapiOptionImplemented OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This attribute, when true, shall indicate that the WAPI option is implemented; else, this attribute shall be false."

::= { gb15629dot11wapiConfigEntry 4 }

gb15629dot11wapiPreauthenticationImplemented OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This variable indicates whether the entity supports WAPI Preauthentication. This cannot be TRUE unless gb15629dot11wapiOptionImplemented is TRUE."

::= { gb15629dot11wapiConfigEntry 5 }

gb15629dot11wapiEnabled OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"When this object is set to TRUE, this shall indicate that WAPI is enabled on this entity. The entity will advertise the WAPI Parameter Set information element in its Beacon and Probe Response frames."

::= { gb15629dot11wapiConfigEntry 6 }

gb15629dot11wapiPreauthenticationEnabled OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"When this object is set to TRUE, this shall indicate that WAPI Preauthentication is enabled on this entity. This object requires that dot11WAPI Enabled also be set to TRUE."

::= { gb15629dot11wapiConfigEntry 7 }

gb15629dot11wapiConfigUnicastKeysSupported OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object indicates the number of the USKs that a WAPI entity supports."

::= { gb15629dot11wapiConfigEntry 8 }

gb15629dot11wapiConfigUnicastRekeyMethod OBJECT-TYPE

SYNTAX INTEGER { disabled(1), timeBased(2),

messageBased(3), timemessage-Based(4) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object selects a mechanism for rekeying the WAPI USK. The default is time-based, once per day. Rekeying the USK is only applicable to an entity acting in the AE or ASUE role."

DEFVAL { timeBased }

::= { gb15629dot11wapiConfigEntry 9 }

gb15629dot11wapiConfigUnicastRekeyTime OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The time in seconds after which the WAPI USK shall be refreshed. The timer shall start at the moment the USK was set using the MLME-SETWPIKEYS.request primitive."

DEFVAL { 86400 }

::= { gb15629dot11wapiConfigEntry 10 }

gb15629dot11wapiConfigUnicastRekeyMessages OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "1000 messages"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"A message count (in 1000s of messages) after which the WAPI USK shall be refreshed. The message counter shall start at the moment the USK was set using the MLME-SETWPIKEYS.request primitive and it shall count all messages encrypted using the current USK."

::= { gb15629dot11wapiConfigEntry 11 }

gb15629dot11wapiConfigMulticastCipher OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object indicates the multicast cipher suite that this entity must adopt. The WAPI Parameter Set information element shall adopt the value of this variable, which contains a 3-octet OUI and a one-octet cipher suite identifier."

::= { gb15629dot11wapiConfigEntry 12 }

gb15629dot11wapiConfigMulticastRekeyMethod OBJECT-TYPE

SYNTAX INTEGER { disabled(1), timeBased(2),

messageBased(3), timemessage-Based(4) }

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object selects a mechanism for rekeying the WAPI MSK. The default is time-based, once per day. Rekeying the MSK is only applicable to an entity acting in the AE role."

DEFVAL { timeBased }

::= { gb15629dot11wapiConfigEntry 13 }

gb15629dot11wapiConfigMulticastRekeyTime OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The time in seconds after which the WAPI MSK shall be refreshed. The timer shall start at the moment the MSK was set using the MLME-SETWPIKEYS.request primitive."

DEFVAL { 86400 }

::= { gb15629dot11wapiConfigEntry 14 }

gb15629dot11wapiConfigMulticastRekeyMessages OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "1000 messages"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"A message count (in 1000s of messages) after which the WAPI MSK shall be refreshed. The message counter shall start at the moment the MSK was set using the MLME-SETWPIKEYS.request primitive and it shall count all messages encrypted using the current MSK."

::= { gb15629dot11wapiConfigEntry 15}

gb15629dot11wapiConfigMulticastRekeyStrict OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object signals that the MSK shall be refreshed whenever a STA leaves the BSS that possesses the MSK."

::= { gb15629dot11wapiConfigEntry 16 }

gb15629dot11wapiConfigPSKValue OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(32))

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The PSK for when WAPI in PSK mode is the selected AKM suite. In that case, the BK will obtain its value from this object. This object is logically write-only. Reading this variable shall return unsuccessful status or null or zero."

::= { gb15629dot11wapiConfigEntry 17 }

gb15629dot11wapiConfigPSKPassPhrase OBJECT-TYPE

SYNTAX DisplayString

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The PSK, for when WAPI in PSK mode is the selected AKM suite, is configured by gb15629dot11wapiConfigPSKValue. An alternative manner of setting the PSK uses the password-to-key algorithm. This variable provides a means to enter a pass-phrase. When this object is written, the WAPI entity shall use the password-to-key algorithm to derive a preshared and populate gb15629dot11wapiConfigPSKValue with this key. This object is logically write-only. Reading this variable shall return unsuccessful status or null or zero."

::= { gb15629dot11wapiConfigEntry 18 }

gb15629dot11wapiConfigCertificateUpdateCount OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The number of times messages in the WAPI Certificate Authentication Handshake protocol will be retried per Certificate Authentication Handshake attempt."

DEFVAL { 3 }

::= { gb15629dot11wapiConfigEntry 19 }

gb15629dot11wapiConfigMulticastUpdateCount OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The number of times Message 1 in the WAPI Multicast Key Announcement Handshake will be retried per MSK Handshake attempt."

DEFVAL { 3 }

::= { gb15629dot11wapiConfigEntry 20 }

gb15629dot11wapiConfigUnicastUpdateCount OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The number of times Message 1 and Message 3 in the WAPI Unicast Key Negotiation Handshake will be retried per USK Handshake attempt."

DEFVAL { 3 }

::= { gb15629dot11wapiConfigEntry 21 }

gb15629dot11wapiConfigMulticastCipherSize OBJECT-TYPE

SYNTAX Unsigned32 (0..4294967295)

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object indicates the length in bit of the MSK. This should be 256 for in SMS4. The first 128bits is the MEK and the last 128bits is the MCK."

::= { gb15629dot11wapiConfigEntry 22 }

gb15629dot11wapiConfigBKLifetime OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "seconds"

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"The maximum lifetime of a BK in the BK cache."

DEFVAL { 43200 }

::= { gb15629dot11wapiConfigEntry 23 }

gb15629dot11wapiConfigBKReauthThreshold OBJECT-TYPE

SYNTAX Unsigned32 (1..100)

UNITS “percentage”

MAX-ACCESS read-write

STATUS current

DESCRIPTION

“The percentage of the BK lifetime that should expire before a WAI reauthentication occurs.”

DEFVAL { 70 }

::= { gb15629dot11wapiConfigEntry 24 }

gb15629dot11wapiConfigSATimeout OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS “seconds”

MAX-ACCESS read-write

STATUS current

DESCRIPTION

“The maximum time a security association shall take to set up.”

DEFVAL { 60 }

::= { gb15629dot11wapiConfigEntry 25 }

gb15629dot11wapiAuthenticationSuiteSelected OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

“The selector of the last AKM suite negotiated.”

::= { gb15629dot11wapiConfigEntry 26 }

gb15629dot11wapiUnicastCipherSelected OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The selector of the last unicast cipher suite negotiated."

::= { gb15629dot11wapiConfigEntry 27 }

gb15629dot11wapiMulticastCipherSelected OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The selector of the last multicast cipher suite negotiated"

::= { gb15629dot11wapiConfigEntry 28 }

gb15629dot11wapiBKIDUsed OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(16))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The selector of the last BKID used in the last Unicast Key Negotiation Handshake."

::= { gb15629dot11wapiConfigEntry 29 }

gb15629dot11wapiAuthenticationSuiteRequested OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-only

STATUS current

```
            DESCRIPTION
                "The selector of the last AKM suite requested."
        ::= { gb15629dot11wapiConfigEntry 30 }

    gb15629dot11wapiUnicastCipherRequested OBJECT-TYPE
            SYNTAX OCTET STRING (SIZE(4))
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
                "The selector of the last unicast cipher suite requested"
        ::= { gb15629dot11wapiConfigEntry 31 }

    gb15629dot11wapiGroupCipherRequested OBJECT-TYPE
            SYNTAX OCTET STRING (SIZE(4))
            MAX-ACCESS read-only
            STATUS current
            DESCRIPTION
                "The selector of the last multicast cipher suite requested."
        ::= { gb15629dot11wapiConfigEntry 32 }
--**********************************************************************
-- *  End of  gb15629dot11wapiConfig table
-- **********************************************************************



-- **********************************************************************
-- *  gb15629dot11wapiConfigUnicastCiphers table
-- **********************************************************************
    gb15629dot11wapiConfigUnicastCiphers OBJECT-TYPE
            SYNTAX SEQUENCE OF gb15629dot11wapiConfigUnicastCiphersEntry
```

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table lists the unicast cipher suites supported by this entity. It allows enabling and disabling of each unicast cipher suite by network management. The unicast cipher suite list in the WAPI Parameter Set information element is formed using the information in this table."

::= { wapiMIBObjects 2 }

gb15629dot11wapiConfigUnicastCiphersEntry OBJECT-TYPE

SYNTAX gb15629dot11wapiConfigUnicastCiphersEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"The table entry, indexed by the interface index (or all interfaces) and the unicast cipher suite."

INDEX { gb15629dot11wapiConfigIndex, gb15629dot11wapiConfigUnicastCipherIndex }

::= { gb15629dot11wapiConfigUnicastCiphersTable 1 }

gb15629dot11wapiConfigUnicastCiphersEntry ::=

SEQUENCE {

gb15629dot11wapiConfigUnicastCipherIndex Unsigned32,

gb15629dot11wapiConfigUnicastCipher OCTET STRING,

gb15629dot11wapiConfigUnicastCipherEnabled TruthValue,

gb15629dot11wapiConfigUnicastCipherSize Unsigned32 }

gb15629dot11wapiConfigUnicastCipherIndex OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

" The auxiliary index into the gb15629dot11wapiConfigUnicastCiphersTable."

::= { gb15629dot11wapiConfigUnicastCiphersEntry 1 }

```
    gb15629dot11wapiConfigUnicastCipher OBJECT-TYPE

        SYNTAX OCTET STRING (SIZE(4))

        MAX-ACCESS read-only

        STATUS current

        DESCRIPTION

            "The selector of a supported unicast cipher suite. It consists of an OUI (the first 3 octets) and a
            cipher suite identifier (the last octet)."

      ::= { gb15629dot11wapiConfigUnicastCiphersEntry 2 }


    gb15629dot11wapiConfigUnicastCipherEnabled OBJECT-TYPE

          SYNTAX TruthValue

          MAX-ACCESS read-write

          STATUS current

          DESCRIPTION

              "This object enables or disables the unicast cipher."

      ::= { gb15629dot11wapiConfigUnicastCiphersEntry 3 }


    gb15629dot11wapiConfigUnicastCipherSize OBJECT-TYPE

          SYNTAX Unsigned32 (0..4294967295)

          MAX-ACCESS read-only

          STATUS current

          DESCRIPTION

              "This object indicates the length in bit of the USK. This should be 256 for SMS4. The first 128bits
              is the UEK and the last 128bits is the UCK."

      ::= { gb15629dot11wapiConfigUnicastCiphersEntry 4 }
-- ************************************************************************

-- *  End of gb15629dot11wapiConfigUnicastCiphers

-- ************************************************************************
```

```
-- ********************************************************************
-- *  gb15629dot11wapiConfigAuthenticationSuites table
-- ********************************************************************
   gb15629dot11wapiConfigAuthenticationSuites OBJECT-TYPE
         SYNTAX SEQUENCE OF gb15629dot11wapiConfigAuthenticationSuitesEntry
         MAX-ACCESS not-accessible
         STATUS current
         DESCRIPTION
             "This table lists the AKM suites supported by this entity. Each AKM suite can be individually
             enabled and disabled. The AKM suite list in the WAPI Parameter Set information element is
             formed using the information in this table."
      ::= { wapiMIBObjects 3 }

   gb15629dot11wapiConfigAuthenticationSuitesEntry OBJECT-TYPE
         SYNTAX gb15629dot11wapiConfigAuthenticationSuitesEntry
         MAX-ACCESS not-accessible
         STATUS current
         DESCRIPTION
             "An entry in the gb15629dot11WAPIConfigAuthenticationSuitesTable."
         INDEX { gb15629dot11wapiConfigAuthenticationSuiteIndex }
      ::= { gb15629dot11wapiConfigAuthenticationSuitesTable 1 }

   gb15629dot11wapiConfigAuthenticationSuitesEntry ::=
         SEQUENCE {
         gb15629dot11wapiConfigAuthenticationSuiteIndex        Unsigned32,
         gb15629dot11wapiConfigAuthenticationSuite             OCTET STRING,
         gb15629dot11wapiConfigAuthenticationSuiteEnabled      TruthValue }

   gb15629dot11wapiConfigAuthenticationSuiteIndex OBJECT-TYPE
         SYNTAX Unsigned32 (1..4294967295)
```

```
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "The auxiliary variable used as an index into the
             gb15629dot11WAPIConfigAuthenticationSuitesTable."
    ::= { gb15629dot11wapiConfigAuthenticationSuitesEntry 1 }

gb15629dot11wapiConfigAuthenticationSuite OBJECT-TYPE
        SYNTAX OCTET STRING (SIZE(4))
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "The selector of an AKM suite. It consists of an OUI (the first 3 octets) and a cipher suite
            identifier (the last octet)."
    ::= { gb15629dot11wapiConfigAuthenticationSuitesEntry 2 }

gb15629dot11wapiConfigAuthenticationSuiteEnabled OBJECT-TYPE
        SYNTAX TruthValue
        MAX-ACCESS read-write
        STATUS current
        DESCRIPTION
            "This variable indicates whether the corresponding AKM suite is enabled/disabled."
    ::= { gb15629dot11wapiConfigAuthenticationSuitesEntry 3 }
-- ****************************************************************
-- *  End of gb15629dot11wapiConfigAuthenticationSuites
-- ****************************************************************


-- ****************************************************************
-- *  gb15629dot11wapiStats table
```

```
-- ****************************************************************

   gb15629dot11wapiStats OBJECT-TYPE

         SYNTAX SEQUENCE OF gb15629dot11wapiStatsEntry

         MAX-ACCESS not-accessible

         STATUS current

         DESCRIPTION

             "This table maintains per-STA statistics in a WAPI. The entry with
             gb15629dot11wapiStatsSTAAddress set to FF-FF-FF-FF-FF-FF shall contain statistics for
             broadcast/multicast traffic."

      ::= { wapiMIBObjects 4 }


   gb15629dot11wapiStatsEntry OBJECT-TYPE

         SYNTAX gb15629dot11wapiStatsEntry

         MAX-ACCESS not-accessible

         STATUS current

         DESCRIPTION

             "An entry in the gb15629dot11wapiStatsTable."

             INDEX { gb15629dot11wapiConfigIndex,gb15629dot11wapiStatsIndex }

      ::= { gb15629dot11wapiStats 1 }


   gb15629dot11wapiStatsEntry ::=

         SEQUENCE {

          gb15629dot11wapiStatsIndex                    Unsigned32,

          gb15629dot11wapiStatsSTAAddress               MacAddress,

          gb15629dot11wapiStatsVersion                  Unsigned32,

          gb15629dot11wapiStatsControlledPortStatus     TruthValue,

          gb15629dot11wapiStatsSelectedUnicastCipher    OCTET STRING,

          gb15629dot11wapiStatsWPIReplayCounters        Counter32,

          gb15629dot11wapiStatsWPIDecryptableErrors     Counter32,

          gb15629dot11wapiStatsWPIMICErrors             Counter32,
```

```
        gb15629dot11wapiStatsWAISignatureErrors                 Counter32,
        gb15629dot11wapiStatsWAIHMACErrors                      Counter32,
        gb15629dot11wapiStatsWAIAuthenticationResultFailures    Counter32,
        gb15629dot11wapiStatsWAIDiscardCounters                 Counter32,
        gb15629dot11wapiStatsWAITimeoutCounters                 Counter32,
        gb15629dot11wapiStatsWAIFormatErrors                    Counter32,
        gb15629dot11wapiStatsWAICertificateHandshakeFailures    Counter32,
        gb15629dot11wapiStatsWAIUnicastHandshakeFailures        Counter32,
        gb15629dot11wapiStatsWAIMulticastHandshakeFailures      Counter32}

gb15629dot11wapiStatsIndex OBJECT-TYPE
        SYNTAX Unsigned32 (1..4294967295)
        MAX-ACCESS not-accessible
        STATUS current
        DESCRIPTION
            "An auxiliary index into the gb15629dot11wapiStatsTable."
    ::= { gb15629dot11wapiStatsEntry 1 }

gb15629dot11wapiStatsSTAAddress OBJECT-TYPE
        SYNTAX MacAddress
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
            "STA's MAC address."
    ::= { gb15629dot11wapiStatsEntry 2 }

gb15629dot11wapiStatsVersion OBJECT-TYPE
        SYNTAX Unsigned32 (1..4294967295)
        MAX-ACCESS read-only
```

STATUS current

DESCRIPTION

"The WAPI version with which the STA associated."

::= { gb15629dot11wapiStatsEntry 3 }

gb15629dot11wapiStatsControlledPortStatus OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This attribute indicates the state of the controlled port in an AE subsystem. If value is "1", the state is set to"authenticated"; else, the state is set to "unauthenticated"."

::= { gb15629dot11wapiStatsEntry 4 }

gb15629dot11wapiStatsSelectedUnicastCipher OBJECT-TYPE

SYNTAX OCTET STRING (SIZE(4))

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The unicast cipher suite selector used during association."

::= { gb15629dot11wapiStatsEntry 5 }

gb15629dot11wapiStatsWPIReplayCounter OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of received WPI MPDUs discarded by the replay mechanism."

::= { gb15629dot11wapiStatsEntry 6 }

gb15629dot11wapiStatsWPIDecryptableErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of received MPDUs discarded by the WPI decryption algorithm because of no valid key."

::= { gb15629dot11wapiStatsEntry 7 }

gb15629dot11wapiStatsWPIMICErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The number of received MPDUs discarded because of the MIC check error during the WPI decryption."

::= { gb15629dot11wapiStatsEntry 8 }

gb15629dot11wapiStatsWAISignatureErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the signature in the received WAI message is incorrect."

::={ gb15629dot11wapiStatsEntry 9 }

gb15629dot11wapiStatsWAIHMACErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the message authentication code in the received WAI message is incorrect."

::={ gb15629dot11wapiStatsEntry 10}

gb15629dot11wapiStatsWAIAuthenticationResultFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the WAI authentication is unsuccessful."

::={ gb15629dot11wapiStatsEntry 11}

gb15629dot11wapiStatsWAIDiscardCounters OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the received WAI message is discarded."

::={ gb15629dot11wapiStatsEntry 12}

gb15629dot11wapiStatsWAITimeoutCounters OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the WAI message is timeout."

::={ gb15629dot11wapiStatsEntry 13}

gb15629dot11wapiStatsWAIFormatErrors OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when there exists format error in the WAI message."

::={ gb15629dot11wapiStatsEntry 14}

gb15629dot11wapiStatsWAICertificateHandshakeFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the WAI Certificate Authentication is unsuccessful."

::={ gb15629dot11wapiStatsEntry 15}

gb15629dot11wapiStatsWAIUnicastHandshakeFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the WAI Unicast Key Negotiation is unsuccessful."

::={ gb15629dot11wapiStatsEntry 16}

gb15629dot11wapiStatsWAIMulticastHandshakeFailures OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This counter shall increment when the WAI Unicast Key Negotiation is unsuccessful."

```
      ::={ gb15629dot11wapiStatsEntry 17}
-- ********************************************************************
-- *  End of gb15629dot11wapiStats
-- ********************************************************************



-- ****************************************
-- *  Conformance information -- gb15629.11 WAPI MIB
-- ****************************************
gb15629dot11wapiGroups OBJECT IDENTIFIER ::= { wapiMIBConformance 1 }
gb15629wapiCompliances OBJECT IDENTIFIER::= { wapiMIBConformance 2 }

-- ********************************************************************
-- * Conformance information – WAPI
-- ********************************************************************

   gb15629dot11wapiCompliance MODULE-COMPLIANCE
           STATUS current
           DESCRIPTION
                "Implement conformance statement of the SNMPv2 entity in the WAPI MIB."
           MODULE
           MANDATORY-GROUPS {
           gb15629dot11wapiBase }
           -- OPTIONAL-GROUPS {gb15629dot11wapiBKcachingGroup }
      ::= { gb15629WapiCompliances 1 }

-- ********************************************************************
-- * Groups - units of conformance– WAPI
-- ********************************************************************
   gb15629dot11wapiBase OBJECT-GROUP
```

OBJECTS {

gb15629dot11wapiConfigVersion,

gb15629dot11wapiControlledAuthControl,

gb15629dot11wapiControlledPortControl,

gb15629dot11wapiOptionalImplemented,

gb15629dot11wapiPreauthenticationImplemented,

gb15629dot11wapiEnabled,

gb15629dot11wapiPreauthenticationEnabled,

gb15629dot11wapiConfigUnicastKeysSupported,

gb15629dot11wapiConfigUnicastRekeyMethod,

gb15629dot11wapiConfigUnicastRekeyTime,

gb15629dot11wapiConfigUnicastRekeyMessages,

gb15629dot11wapiConfigMulticastCipher,

gb15629dot11wapiConfigMulticastRekeyMethod,

gb15629dot11wapiConfigMulticastRekeyTime,

gb15629dot11wapiConfigMulticastRekeyMessages,

gb15629dot11wapiConfigMulticastRekeyStrict,

gb15629dot11wapiConfigPSKValue,

gb15629dot11wapiConfigPSKPassPhrase,

gb15629dot11wapiConfigCertificateUpdateCount,

gb15629dot11wapiConfigMulticastUpdateCount,

gb15629dot11wapiConfigUnicastUpdateCount,

gb15629dot11wapiConfigMulticastCipherSize,

gb15629dot11wapiConfigUnicastCipher,

gb15629dot11wapiConfigUnicastCipherEnabled,

gb15629dot11wapiConfigUnicastCipherSize,

gb15629dot11wapiConfigAuthenticationSuite,

gb15629dot11wapiConfigAuthenticationSuiteEnabled,

gb15629dot11wapiConfigSATimeout,

gb15629dot11wapiAuthenticationSuiteSelected,

gb15629dot11wapiUnicastCipherSelected,

gb15629dot11wapiMulticastCipherSelected,

gb15629dot11wapiBKIDUsed,

gb15629dot11wapiAuthenticationSuiteRequested,

gb15629dot11wapiUnicastCipherRequested,

gb15629dot11wapiMulticastCipherRequested,

gb15629dot11wapiStatsSTAAddress,

gb15629dot11wapiStatsVersion,

gb15629dot11wapiStatsControlledPortStatus,

gb15629dot11wapiStatsSelectedUnicastCipher,

gb15629dot11wapiStatsWPIReplayCounters,

gb15629dot11wapiStatsWPIDecryptableErrors,

gb15629dot11wapiStatsWPIMICErrors,

gb15629dot11wapiStatsWAISignatureErrors,

gb15629dot11wapiStatsWAIHMACErrors,

gb15629dot11wapiStatsWAIAuthenticationResultFailures,

gb15629dot11wapiStatsWAIDiscardCounters,

gb15629dot11wapiStatsWAITimeoutCounters,

gb15629dot11wapiStatsWAIFormatErrors,

gb15629dot11wapiStatsWAICertificateHandshakeFailures,

gb15629dot11wapiStatsWAIUnicastHandshakeFailures,

gb15629dot11wapiStatsWAIMulticastHandshakeFailures

}

STATUS current

DESCRIPTION

" The gb15629dot11wapiBase class provides the necessary support to manage the WAPI function of STA."

::= { gb15629dot11WapiGroups 28 }

```
    gb15629dot11wapiBKcachingGroup OBJECT-GROUP

            OBJECTS {gb15629dot11wapiConfigBKLifetime,gb15629dot11wapiConfigBKReauthThreshold}

            STATUS current

            DESCRIPTION

                "The gb15629dot11wapiBKcachingGroup class provides the necessary support to manage the
                BK cache function of STA."

        ::= { gb15629dot11WapiGroups 29 }


--*****************************************************************************

--*  End of gb15629dot11-WAPI-MIB

--*****************************************************************************

    END
```

Annex E
(informative)

High Rate PHY/FH interoperability

See Annex F of ISO/IEC 8802-11:2005.

Annex F
(informative)

An example of encoding a frame for OFDM PHY

See Annex G of ISO/IEC 8802-11:2005.

Annex G
(informative)

Reference implementations of the frame authentication algorithm, the key derivation algorithm and the test vectors

G.1 Frame authentication algorithm

G.1.1 Reference implementation

*int hmac_SHA256(unsigned char *text, int text_len, byte *key, unsigned key_len, byte *digest, unsigned digest_length)*

/*

where

a) *unsigned char *text* indicates the text to be calculated by HMAC;

b) *unsigned text_len* indicates the length of the text to be calculated by HMAC (in octet);

c) *byte *key* , indicates the key used in the HMAC computation;

d) *unsigned key_len* indicates the length of the key used in the HMAC computation (in octet);

e) *byte *digest* indicates the output digest of HMAC;

f) *unsigned digest_length* specifies the length of the output digest of HMAC (in octet) which must be less than that of the hash algorithm SHA256 (in octet);

g) the returned value. The non-zero values indicate the practical length of the output digest, and a value of 0 indicates a failure.

*/

{

byte real_key[SHA256_BLOCK_SIZE];

byte ipad[SHA256_BLOCK_SIZE];

byte opad[SHA256_BLOCK_SIZE];

byte temp_digest1[SHA256_DIGEST_SIZE];

byte temp_digest2[SHA256_DIGEST_SIZE];

CONTX input_data[2];

unsigned i;

```
if digest_length>SHA256_DIGEST_SIZE)
    return 0;

for(i=0;i< SHA256_BLOCK_SIZE;i++){
    real_key[i]=0;
    ipad[i]=0x36;
    opad[i]=0x5c;
}

/* if key_len is larger than hash block size, key is hashed first to make its length is equal hash block size
*/
if(key_len> SHA256_BLOCK_SIZE){
    input_data[0].buff=key;
    input_data[0].length=key_len;
    SHA256(input_data,1,real_key);
    key_len= SHA256_ BLOCK _SIZE;
}
else
    memcpy(real_key,key,key_len);

for(i=0;i< SHA256_BLOCK_SIZE;i++){
    ipad[i]^=real_key[i];
    opad[i]^=real_key[i];
}
```

```
    /*SHA256(Key xor ipad,text)=temp_digest1 */
    input_data[0].buff=ipad;
    input_data[0].length= SHA256_BLOCK_SIZE;
    input_data[1].buff=text;
    input_data[1].length=text_len;
    SHA256(input_data,2,temp_digest1);

    /*SHA256(Key xor opad,temp_digest1)=temp_digest2 */
    input_data[0].buff=opad;
    input_data[0].length= SHA256_BLOCK_SIZE;
    input_data[1].buff=temp_digest1;
    input_data[1].length SHA256_DIGEST_SIZE;
    SHA256(input_data,2,temp_digest2);

    /*output the digest of required length */
    memcpy(digest,temp_digest2,digest_length);
    return digest_length;
}
```

G.1.2 Test vectors

Test vector 1	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20
Length of key	32
Data	**abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopqabcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq** 0x61 62 63 64 62 63 64 65 63 64 65 66 64 65 66 67 65 66 67 68 66 67 68 69 67 68 69 6a 68 69 6a 6b 69 6a 6b 6c 6a 6b 6c 6d 6b 6c 6d 6e 6c 6d 6e 6f 6d 6e 6f 70 6e 6f 70 71 61 62 63 64 62 63 64 65 63 64 65 66 64 65 66 67 65 66 67 68 66 67 68 69 67 68 69 6a 68 69 6a 6b 69 6a 6b 6c 6a 6b 6c 6d 6b 6c 6d 6e 6c 6d 6e 6f 6d 6e 6f 70 6e 6f 70 71

Length of data	112
Digest	0x47 03 05 fc 7e 40 fe 34 d3 ee b3 e7 73 d9 5a ab 73 ac f0 fd 06 04 47 a5 eb 45 95 bf 33 a9 d1 a3

Test vector 2	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
Length of key	37
Data	0xcd repeated by 50 times
Length of data	50
Digest	0xd4 63 3c 17 f6 fb 8d 74 4c 66 de e0 f8 f0 74 55 6e c4 af 55 ef 07 99 85 41 46 8e b4 9b d2 e9 17

Test vector 3	
Key	0x0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b 0b
Length of key	32
Data	**Hi There** 0x48 69 20 54 68 65 72 65
Length of data	8
Digest	0x19 8a 60 7e b4 4b fb c6 99 03 a0 f1 cf 2b bd c5 ba 0a a3 f3 d9 ae 3c 1c 7a 3b 16 96 a0 b6 8c f7

Test vector 4	
Key	0x4a 65 66 65
Length of key	4
Data	**what do ya want for nothing?** 0x77 68 61 74 20 64 6f 20 79 61 20 77 61 6e 74 20 66 6f 72 20 6e 6f 74 68 69 6e 67 3f
Length of data	28

Digest	0x5b dc c1 46 bf 60 75 4e 6a 04 24 26 08 95 75 c7 5a 00 3f 08 9d 27 39 83 9d ec 58 b9 64 ec 38 43

G.2 Key derivation algorithm

G.2.1 Reference implementation

void KD_hmac_SHA256(byte *text,unsigned text_len,byte *key,unsigned key_len,byte *output,unsigned length)

/*

where

a) *byte *text* indicates the input text of the key derivation algorithm;

b) *unsigned text_len* indicates the length of the input text (in octet);

c) *byte *key* indicates the input key of the key derivation algorithm;

d) *unsigned key_len* indicates the length of the input key (in octet);

e) *byte *output* indicates the output of the key derivation algorithm;

f) *unsigned length* indicates the length of the output of the key derivation algorithm (in octet).

/

{

```
    int i;
    for(i=0;length/SHA256_DIGEST_SIZE;i++,length-=SHA256_DIGEST_SIZE){
            hmac(MHASH_SHA256,text,text_len,key,key_len,&output[i*SHA256_DIGEST_SIZE]
        ,SHA256_DIGEST_SIZE);
        text=&output[i*SHA256_DIGEST_SIZE];
        text_len=SHA256_DIGEST_SIZE;
    }
    if(length>0)
        hmac(MHASH_SHA256,text,text_len,key,key_len,&output[i*SHA256_DIGEST_SIZE],length);
}
```

G.2.2 Test vectors

Test vector 1	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20
Length of key	32
Data	**pairwise key expansion for infrastructure unicast** 0x70 61 69 72 77 69 73 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 69 6e 66 72 61 73 74 72 75 63 74 75 72 65 20 75 6e 69 63 61 73 74
Length of data	49
Output	0xe3 a6 45 46 f2 d1 f5 ee b7 d1 ee 06 d2 c9 e5 4a 2c c9 d6 ce c3 b7 6f fd 62 63 f4 26 dc 25 39 af bd 98 80 a5 27 a1 b5 85 59 4b 57 ce 33 21 4f 0c
Length of output	48

Test vector 2	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
Length of key	37
Data	**pairwise key expansion for infrastructure unicast** 0x70 61 69 72 77 69 73 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 69 6e 66 72 61 73 74 72 75 63 74 75 72 65 20 75 6e 69 63 61 73 74
Length of data	49
Output	0x3b 6e ca 4f 08 76 c4 3a b3 1b 26 3f 2c 38 b8 81 21 b5 68 e5 f8 fd 1d 4c fa 4c 7f 8c 60 97 04 3d 7b 40 a8 63 b9 43 b9 f5 bb 37 2f 3a dd a5 da 27
Length of output	48

Test vector 3	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15
Length of key	16

Data	**pairwise key expansion for infrastructure unicast** 0x70 61 69 72 77 69 73 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 69 6e 66 72 61 73 74 72 75 63 74 75 72 65 20 75 6e 69 63 61 73 74
Length of data	49
Output	0xbc 29 f3 e6 09 1f 6a c9 0b a0 20 61 92 12 48 69 5f ee ff 1a 4c ab 53 3b 11 67 d8 54 5f 93 5f 28 11 84 c9 bb 32 f9 87 b9 86 81 0f fb 17 c4 10 f5
Length of output	48

Test vector 4	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20
Length of key	32
Data	**group key expansion for multicast and broadcast** 0x67 72 6f 75 70 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 6d 75 6c 74 69 63 61 73 74 20 61 6e 64 20 62 72 6f 61 64 63 61 73 74
Length of data	47
Output	0x20 8f 72 54 a4 bf 56 f0 fa 49 5f e1 0c 99 15 05 92 ed 79 df 57 74 a9 6e 13 97 1e c4 a1 5e 16 a7 ed 75 f5 e5 44 bb d3 35 67 eb 88 e7 83 24 a9 d2
Length of output	48

Test vector 5	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
Length of key	37
Data	**group key expansion for multicast and broadcast** 0x67 72 6f 75 70 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 6d 75 6c 74 69 63 61 73 74 20 61 6e 64 20 62 72 6f 61 64 63 61 73 74
Length of data	47
Output	0x33 32 61 7a 90 8e a5 a0 7f fa 1d 23 79 f3 d8 3e 8b e9 14 1f 15 53 8f d3 ef de 58 01 19 e8 c5 09 5d 25 b2 d3 0a c7 a6 35 ad b4 3c 6c ac f0 aa 2b

Length of output	48

Test vector 6	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15
Length of key	16
Data	**group key expansion for multicast and broadcast** 0x67 72 6f 75 70 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 6d 75 6c 74 69 63 61 73 74 20 61 6e 64 20 62 72 6f 61 64 63 61 73 74
Length of data	47
Output	0xf2 cb f1 1c 6d 40 b8 09 d0 c0 ed 48 2a 4a 1b 6a 15 1a f1 fb 4c 80 f9 80 5c 93 e5 6e b1 cf 5c b5 ec c1 3e 7a bc af e0 a7 d2 59 5d 51 9b 76 9a 24
Length of output	48

Test vector 7	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20
Length of key	32
Data	**pre-share key expansion for adhoc network** 0x70 72 65 2d 73 68 61 72 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 61 64 68 6f 63 20 6e 65 74 77 6f 72 6b
Length of data	41
Output	0xc0 7a d8 32 25 2a 0c 14 76 18 f4 c0 d0 6b 35 f4 f6 d6 73 5d 1a a3 8e 47 9a 7e e0 ac 1c 0c 38 5b 2d 33 28 74 1e 4d a0 c8 76 fc 6c c9 e3 60 c8 d7
Length of output	48

Test vector 8	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25
Length of key	37
Data	**pre-share key expansion for adhoc network** 0x70 72 65 2d 73 68 61 72 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 61 64 68 6f 63 20 6e 65 74 77 6f 72 6b
Length of data	41
Output	0x f0 0b ee f2 f5 5f 85 d8 ee b0 6f 8c c4 1b e6 0e c2 69 f5 82 9a 0b 6e fb 2d 9b 49 5e b1 87 d3 58 59 68 88 c3 d2 6f 94 9f 8d 2e 41 fe bc bb b9 9a
Length of output	48

Test vector 9	
Key	0x01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15
Length of key	16
Data	**pre-share key expansion for adhoc network** 0x70 72 65 2d 73 68 61 72 65 20 6b 65 79 20 65 78 70 61 6e 73 69 6f 6e 20 66 6f 72 20 61 64 68 6f 63 20 6e 65 74 77 6f 72 6b
Length of data	41
Output	0x05 8e b8 7c ff 82 66 47 de 50 7b 14 17 ac 99 6e b5 7f cf 11 fd fc 83 be 59 d5 85 f4 a7 3e 69 7d d4 38 e3 34 fe bb 06 7d 14 6f 01 31 a6 96 4f 26
Length of output	48

Annex H
(informative)

Examples of WAI parameters and WPI block cryptographic algorithm

H.1 Principle

Cryptographic algorithms and ECC parameters to be applied to information security mechanism may be subject to national and regional regulations. They should conform to national laws and regulations, and can be chosen according to specific requirements in different countries and regions. An appropriate block algorithm should be adopted in WPI, however the type of algorithm to be adopted is optional. The appropriate ECC parameters should be adopted in WAI, however the parameter to be adopted is optional.

An appropriate algorithm means that this algorithm should meet the requirements concerning the block size and the key length, 128 bits.

H.2 Algorithm used in China

In this part of ISO/IEC 8802, specific cryptographic algorithms are instances of implementation. Cryptographic algorithms to be applied in WPI may be subject to national and regional regulations, and they should have a 128-bit block size and a 128-bit key length. In China, the adopted cryptographic algorithm is SMS4, which has a 128-bit block size and a 128-bit key length. The corresponding OUI of SMS4 is 00-14-72 and the Suite Type is 1.

Detailed information concerning SMS4 can be obtained from the website [Chinese]http://www.oscca.gov.cn or [English]http://eprint.iacr.org/2008/329.pdf.

H.3 ECC parameters used in China

In this part of ISO/IEC 8802, no specific ECC parameters are defined. ECC parameters to be applied in WAI may be subject to national and regional regulations.

The OID corresponding to ECC parameters chosen in China is 1.2.156.11235.1.1.2.1. The OID 1.2.156.11235.1.1.2.1 is also applied in GBW and ECDH. Detailed information can be obtained from the website http://www.oscca.gov.cn.

Bibliography

See Annex E of ISO/IEC 8802-11:2005, Amd 4:2006, Amd 5:2006 and Amd 6:2006, and also:

[1] ISO/IEC 7498-1, *Information technology — Open Systems Interconnection — Basic Reference Model: The Basic Model*

[2] ISO/IEC 8802-2, *Information technology — Telecommunications and information exchange between systems—Local and metropolitan area networks — Specific requirements — Part 2: Logical link control*

[3] ISO/IEC 8824-1:2002, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation*

[4] ISO/IEC 8824-2:2002, *Information technology — Abstract Syntax Notation One (ASN.1): Information object specification*

[5] ISO/IEC 8824-3:2002, *Information technology — Abstract Syntax Notation One (ASN.1): Constraint specification*

[6] ISO/IEC 8824-4:2002, *Information technology — Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*

[7] ISO/IEC 8825-1:2002, *Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*

[8] ISO/IEC 8825-2:2002, *Information technology — ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*

[9] ISO/IEC 15802-1:2002, *Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Common specifications — Part 1: Medium Access Control (MAC) service definition*

[10] ISO/IEC 15946, *Information technology — Security techniques — Cryptographic techniques based on elliptic curves*

[11] IEEE Std C95.1-2005, *IEEE Standard Safety Levels with Respect to Human Exposure to Radio Frequency Electromagnetic Fields, 3 kHz to 300 GHz*

[12] IETF RFC 1750, *Randomness Recommendations for Security*, December 1994

[13] IETF RFC 2104, *HMAC: Keyed-Hashing for Message Authentication*

[14] IETF RFC 3610, *Counter with CBC-MAC (CCM),* September 2003

[15] IETF RFC 3748, *Extensible Authentication Protocol (EAP),* June 2004

[16] ITU Radio Regulations, Volumes 1–4

[17] ITU-T Recommendation 210 (11/93), *Information technology — Open systems interconnection — Basic Reference Model: Conventions for the definition of OSI services* (identical to ISO/IEC 10731)

[18] ITU-T Recommendation X.509, ISO/IEC 9594-8:2005, *Information technology — Open Systems Interconnection—The Directory: Public-key and attribute certificate frameworks* (identical to ISO/IEC 9594-8)

[19] ITU-T Recommendation Z.100 (08/02), *CCITT specification and description language (SDL)*

[20] ITU-T Recommendation Z.105 (07/03), *SDL combined with ASN.1 modules (SDL/ASN.1)*

[21] *SMS4 Encryption Algorithm for Wireless Networks* http://eprint.iacr.org/2008/329.pdf