IEEE STANDARDS ASSOCIATION

IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS)

IEEE Vehicular Technology Society

Sponsored by the
Intelligent Transportation Systems (VT/ITS) Committee

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

IEEE Std 1609.11™-2010

9 January 2011

IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS)

Sponsor

Intelligent Transportation Systems (VT/ITS) Committee
of the
IEEE Vehicular Technology Society

Approved 9 November 2010
IEEE-SA Standards Board

Abstract: This standard specifies the electronic payment service layer and profile for Payment and Identity authentication, and Payment Data transfer for Dedicated Short Range Communication (DSRC) based applications in Wireless Access in Vehicular Environments. This standard defines a basic level of technical interoperability (vehicle-to-roadside) for electronic payment equipment, i.e., onboard unit (OBU) and roadside unit (RSU) using WAVE. It does not provide a full solution for interoperability, and it does not define other parts of the electronic payment system, other services, other technologies and non-technical elements of payment interoperability. This standard is not intended to define technology and processes to activate and store data into the OBU (personalization), nor the applications using the payment service.
Keywords: EFC, electronic fee collection, electronic payment service, EPS, WAVE, Wireless Access in Vehicular Environments

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

Copyright © 2011 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 9 January 2011. Printed in the United States of America.

IEEE and 802 are registered trademarks in the U.S. Patent & Trademark Office, owned by the Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-0-7381-6501-1 STD97049
Print: ISBN 978-0-7381-6502-8 STDPD97049

IEEE prohibits discrimination, harassment and bullying. For more information, visit http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html.
No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

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

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

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

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

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

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration. A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal interpretation of the IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position, explanation, or interpretation of the IEEE.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Recommendations to change the status of a stabilized standard should include a rationale as to why a revision or withdrawal is required. Comments and recommendations on standards, and requests for interpretations should be addressed to:

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

Authorization to photocopy portions of any individual standard for internal or personal use is granted by The Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

This introduction is not part of IEEE Std 1609.11-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS).

This standard gives requirements for services for the handling of electronic payment information in WAVE systems, as described in the IEEE 1609 family of standards.

Notice to users

Laws and regulations

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

Copyrights

This document is copyrighted by the IEEE. It is made available for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making this document available for use and adoption by public authorities and private users, the IEEE does not waive any rights in copyright to this document.

Updating of IEEE documents

Users of IEEE standards should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect. In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE Standards Association web site at http://ieeexplore.ieee.org/xpl/standards.jsp, or contact the IEEE at the address listed previously.

For more information about the IEEE Standards Association or the IEEE standards development process, visit the IEEE-SA web site at http://standards.ieee.org.

Errata

Errata, if any, for this and all other standards can be accessed at the following URL: http://standards.ieee.org/reading/ieee/updates/errata/index.html. Users are encouraged to check this URL for errata periodically.

Copyright © 2011 IEEE. All rights reserved.

Interpretations

Current interpretations can be accessed at the following URL: http://standards.ieee.org/reading/ieee/interp/index.html.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. A patent holder or patent applicant has filed a statement of assurance that it will grant licenses under these rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses. Other Essential Patent Claims may exist for which a statement of assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this standard was submitted to the IEEE-SA Standards Board for approval, the Dedicated Short Range Communication Working Group had the following membership:

Thomas M. Kurihara, *Chair*
John Moring, *Vice Chair*

Scott Andrews
Roger Berg
Anthony Chou
Susan Dickey
Wayne Fisher
Ramez Gerges
Paul Gray
Refi-Tugrul Güner
Somak Datta Gupta
Gloria Gwynne
Ron Hochnadel
Gavin Holland
Wendong Hu
Guo-Huang Hsu
Daniel Jiang
Carl Kain
Doug Kavner
David Kelley
John Kenney
Minjung Kim
Jerry Landt
Michael Li
Jason Liu
Jules Madey
Alastair Malarky
Justin McNew
Vinuth Rai
Randy Roebuck
Richard Roy
Kuo Sheng-Po
François Simon
George Vlantis
Timothy Weil
Aaron Weinfield
William Whyte

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

Iwan Adhicandra
Wole Akpose
Lee Armstrong
John Barr
H. Stephen Berger
Georgios Bertos
William Byrd
R. Neill Cameron
Kai T. Chen
Keith Chow
Susan Dickey
Carlo Donati
Wayne Fisher
Andre Fournier
Ramez Gerges

Copyright © 2011 IEEE. All rights reserved.

Paul Gray
Randall Groves
Tugrul Guener
Gloria Gwynne
Ronald Hochnadel
Werner Hoelzl
Chung-Hsien Hsu
Wendong Hu
Daniel Jiang
Piotr Karocki
Douglas Kavner
John Kenney
Stuart J. Kerry
Dmitri Khijniak
Max Kicherer
Hsu Kuo-Huang
Thomas M. Kurihara
Paul Lambert
Richard Lancaster
Jeremy Landt
Hsia-Hsin Li
Jun Liu
G. Luri
Dmitriy Lyubskiy
Alastair Malarky
Sean Maschue
Justin McNew
John Moring
Michael S. Newman
Satoshi Oyama
Panagiotis Papadimitratos
Venkatesha Prasad
Robert Robinson
Randal Roebuck
Richard Roy
Randall Safier
Reza Salehi
Bartien Sayogo
Gil Shultz
Walter Struppler
Mark Sturza
Dale Sumida
Jasja Tijink
George Vlantis
Stephen Webb
William Whyte

When the IEEE-SA Standards Board approved this amendment on 9 November 2010, it had the following membership:

Robert M. Grow, *Chair*
Richard H. Hulett, *Vice Chair*
Steve M. Mills, *Past Chair*
Judith Gorman, *Secretary*

Karen Bartleson
Victor Berman
Ted Burse
Clint Chaplin
Andy Drozd
Alexander Gelman
Jim Hughes
Young Kyun Kim
Joseph L. Koepfinger*
John Kulick
David J. Law
Hung Ling
Oleg Logvinov
Ted Olsen
Ronald C. Petersen
Thomas Prevost
Jon Walter Rosdahl
Sam Sciacca
Mike Seavey
Curtis Siller
Don Wright

*Member Emeritus

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

Satish Aggarwal, *NRC Representative*
Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*

Catherine Berger
IEEE Standards Project Editor

Matthew J. Ceglia
IEEE Standards Program Manager, Technical Program Development

Copyright © 2011 IEEE. All rights reserved.

Contents

1. Overview 1
1.1 Scope 1
1.2 Purpose 2
1.3 Document organization 2

2. Normative references 2

3. Definitions, abbreviations, and acronyms 3
3.1 Definitions 3
3.2 Abbreviations and acronyms 4

4. General description 5

5. Payment Service overview 7
5.1 Payment Service functions 7
5.2 MIB maintenance 8

6. Types of systems using EPS 8
6.1 Modes of EPS information exchange 8
6.2 Payer and payee approval types 10

Annex A (normative) Profile for Electronic Fee Collection (EFC) systems per ISO 14906 and ISO 15628 13
A.1 Introduction 13
A.2 General description 13
A.3 WIAL and WAVE services 15
A.4 Interfaces 21
A.5 EFC EPS transactions 39
A.6 Keying material 41

Annex B (informative) Examples of Electronic Fee Collection (EFC) systems per ISO 14906 and ISO 15628 44
B.1 Introduction 44
B.2 Example EFC transaction 44
B.3 Example of an EFC system that uses small zones and un-augmented location determination, with high-speed payers 48

Annex C (normative) Protocol Implementation Conformance Statement (PICS) proforma 52

Copyright © 2011 IEEE. All rights reserved.

IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Over-the-Air Electronic Payment Data Exchange Protocol for Intelligent Transportation Systems (ITS)

IMPORTANT NOTICE: This standard is not intended to ensure safety, security, health, or environmental protection. Implementers of the standard are responsible for determining appropriate safety, security, environmental, and health practices or regulatory requirements.

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

1. Overview

1.1 Scope

This standard specifies the application service layer and profile for Payment and Identity authentication, and Payment Data transfer for Dedicated Short Range Communication (DSRC) based applications using IEEE Std 802.11™ and IEEE 1609 protocols in Wireless Access in Vehicular Environments. This standard defines a basic level of technical interoperability for electronic payment equipment, i.e., onboard unit (OBU) and roadside unit (RSU). It does not provide a full solution for interoperability, and it does not define other parts of the electronic payment-system, other services, other technologies, and non-technical elements of interoperability. This standard is not intended to define technology and processes to activate and store data into the OBU (personalization).

Copyright © 2011 IEEE. All rights reserved.

1.2 Purpose

This standard provides an open standard for the relevant interface in WAVE-based transaction systems, providing a common interoperable service for device identity and payment authentication, and payment data transfer. This allows for interoperability between systems of different vendors and/or operators.

1.3 Document organization

The body of this standard gives electronic payment service (EPS) requirements that are common to all payment systems. Clause 1 provides an overview of the document. Clause 2 and Clause 3 contain references, definitions, and abbreviations, respectively. Clause 4 contains a general description of payment systems. Clause 5 describes a Payment Service, which is the set of EPS functions used by applications. Clause 6 describes types of communication and approval of payment information.

Detailed requirements and profiles for various payment systems are given in annexes to this standard. Annex A contains a description of payment systems whose requirements are given in ISO 14906 and 15628. Annex B contains an example of a system per Annex A. Profiles for other payment systems may be added as annexes as requirements are identified.

Annex C contains the Protocol Implementation Conformance Statement (PICS) proforma.

2. Normative references

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

FIPS Publication 197 (November 2001), Specification for the Advanced Encryption Standard (AES).[1]

IEEE Std 802.11™, IEEE Standard for Local and Metropolitan Area Networks, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.[2, 3]

IEEE Std 802.11p™, IEEE Standard for Local and Metropolitan Area Networks, Amendment 7: Wireless Access in Vehicular Environments.

IEEE P1609.0™ (Draft 0.9, April 2010), Draft Standard for Wireless Access in Vehicular Environments (WAVE)—Architecture.[4]

IEEE P1609.2™ (Draft 5, June 2010), Draft Standard for Wireless Access in Vehicular Environments (WAVE)—Security Services for Applications and Management Messages.

IEEE Std 1609.3™-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Networking Services.

---

[1] Federal Information Processing Standards (FIPS) publications are available from National Technical Information Service (NTIS), 5285 Port Royal Road, Springfield, VA 22161, USA (http://csrc.nist.gov/publications/).
[2] IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (http://standards.ieee.org/).
[3] The IEEE standards or products referred to in this clause are trademarks of the Institute of Electrical and Electronics Engineers, Inc.
[4] Numbers preceded by P are IEEE authorized standards projects that were not approved by the IEEE-SA Standards Board at the time this publication went to Sponsor ballot/press. For information about obtaining drafts, contact the IEEE.

Copyright © 2011 IEEE. All rights reserved.

IEEE Std 1609.4™-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)—Multi-channel Operation.

ISO 14906 Road transport and traffic telematics—Electronic fee collection—Application interface definition for dedicated short-range communication.[5]

ISO 15628 Road transport and traffic telematics—Dedicated short range communication (DSRC)—DSRC application layer.

NIST Special Publication 800-38A 2001, Recommendation for Block Cipher Modes of Operation: Methods and Techniques, National Institute of Standards and Technology.[6]

NIST Special Publication 800-38B 2005 Edition, Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, National Institute of Standards and Technology.

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. *The IEEE Standards Dictionary: Glossary of Terms & Definitions* should be referenced for terms not defined in this clause.[7] Additional WAVE definitions may be found in IEEE P1609.0/D0.9.

application: A higher-layer entity that requires electronic payment service (EPS) functions and communicates via the WAVE stack.

control channel (CCH): A single radio channel, not a service channel, intended for exchange of management information including WAVE service advertisements and WAVE short messages.

Electronic Fee Collection (EFC): A system that is a special case of an EPS system.

Electronic Payment Service (EPS): Pertains to the origination, communication, processing, storage, or retrieval of electronic payment information.

notification: An indication of an event of interest, sent from a management entity to a higher-layer entity.

payee action pre-approval: The payee does not need to wait for approval before it takes further action.

payee action without pre-approval: The payee needs to wait for approval from an external entity before it may take further action.

payer action pre-approval: The payer does not need to wait for approval before taking further action.

payer action without approval: The payer must wait until it receives the payment information confirmation from the payee before taking further action.

[5] ISO publications are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/ Suisse (http://www.iso.ch/). ISO publications are also available in the United States from the Sales Department, American National Standards Institute, 11 West 42nd Street, 13th Floor, New York, NY 10036, USA (http://www.ansi.org/).
[6] National Institute of Standards and Technology (NIST) publications are available from the Superintendent of Documents, U.S. Government Printing Office, Stop SSOP, Washington, DC, USA 20402-0001 (http://bookstore.gpo.gov).
[7] *The IEEE Standards Dictionary: Glossary of Terms & Definitions* is available at http://shop.ieee.org/.

Copyright © 2011 IEEE. All rights reserved.

payer response pre-approval: The payer does not need to wait for approval before it may send the payment information response.

payer response without pre-approval: The payer needs to wait for approval from an external entity before it may send the payment information response.

provider: Advertiser of a WAVE service. *See also:* **user**, **WAVE service advertisement**.

Provider Service Context (PSC): A field associated with a PSID containing supplementary information related to the service. The internal format of the PSC is PSID dependent.

Provider Service Identifier (PSID): An octet string that identifies a service provided by a higher layer entity.

service channel (SCH): Any channel that is not the control channel, intended for management frames and higher layer information exchanges (WSMs and IPv6 packets). There may be more than one service channel defined in a given spectrum.

user: One that acts on receipt of a WAVE service advertisement. *See also:* **provider**, **WAVE service advertisement**.

Vendor Specific Action frame: A management frame specified in IEEE Std 802.11. Despite the name, when the first 36 bits of the Organization Identifier are equal to 0x0050C24A4, the contents are specified within the 1609 standards for carrying IEEE 1609 management information including WAVE service advertisements.

WAVE device: A device that is compliant to IEEE Std 1609.3-2010, IEEE Std 1609.4-2010, and IEEE Std 802.11, operating outside the context of a basic service set (see IEEE P1609.0/D0.9 and IEEE 802.11p).

WAVE management entity (WME): A set of management functions required to provide WAVE networking services. It is specified in IEEE Std 1609.3-2010.

WAVE service advertisement (WSA): A data structure specified in IEEE Std 1609.3-2010 containing information including the announcement of the availability of services

WAVE short message (WSM): A packet consisting of WSM data and a WSMP header.

WAVE short message protocol (WSMP): A protocol for rapid exchange of messages in a rapidly-varying radio frequency (RF) environment where low-latency is also an important objective.

3.2 Acronyms and abbreviations

BST	Beacon Service Table
CBC	cipher-block chaining
CCH	control channel
CMAC	cipher-based message authentication code
DSRC	Dedicated Short Range Communication
EFC	Electronic Fee Collection
EPS	Electronic Payment Service
FIPS	Federal Information Processing Standards

Copyright © 2011 IEEE. All rights reserved.

IP	Internet Protocol
IPv6	Internet Protocol version 6
ITS	Intelligent Transportation Systems
IV	initialization vector
LID	logical link control identifier
MAC	medium access control
MIB	management information base
MLME	MAC sublayer management entity
N/A	not applicable
NIST	National Institute of Standards and Technology
OBU	onboard unit NOTE—In some references, the term onboard equipment (OBE) is used instead of OBU.[8]
PSC	Provider Service Context
PSID	Provider Service Identifier
RF	radio frequency
RSU	roadside unit NOTE—In some references, the term roadside equipment (RSE) is used instead of RSU.
SAP	service access point
SCH	service channel
T-ASDU	transfer-application data unit
UDP	User Datagram Protocol
VST	Vehicle Service Table
WAVE	Wireless Access in Vehicular Environments
WIAL	WAVE Interface Application Layer
WME	WAVE management entity
WSA	WAVE service advertisement
WSM	WAVE short message
WSMP	WAVE short message protocol

4. General description

This standard describes exchanges of electronic payment information between payment equipment units over WAVE. Examples of electronic payment information include the following:

— Account information
— Identity information
— Invoices
— Proof of payment (i.e., receipts)

[8] Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.

Copyright © 2011 IEEE. All rights reserved.

Processing of electronic payment information requires electronic payment service (EPS) functions, which may include the origination, communication, processing, storage or retrieval of the information. EPS functions may also include security tasks such as the encryption or decryption of payment information.

WAVE communications on a payment equipment unit is done using the WAVE protocol stack, as described in IEEE Std 1609.3-2010 and IEEE Std 1609.4-2010. In this standard, an *application* is defined as a higher-layer entity that requires EPS functions and communicates via the WAVE stack.

More than one application on a payment equipment unit may use the same set of EPS functions. For example, a group of applications may be tied to the same account and therefore require common identical payment processing functions.

The set of EPS functions that are used by one or more applications is called the *Payment Service* that serves the application(s).

Figure 1 illustrates the relationships among two applications, which wish to exchange payment information on WAVE, on different equipment units and their respective Payment Services and WAVE stacks.

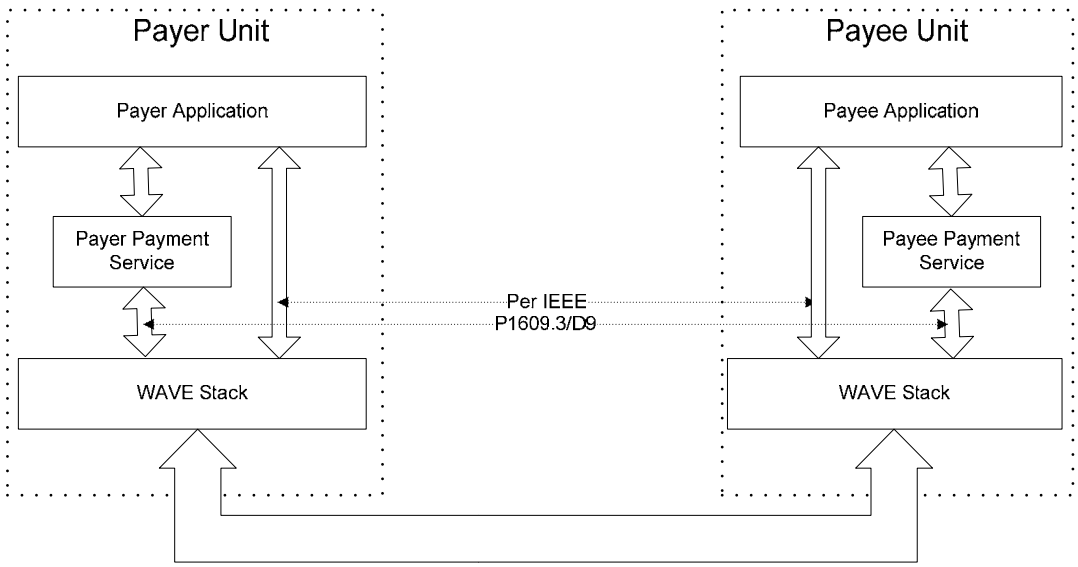

Figure 1—Relationships among applications on different units and their respective Payment Services and WAVE stacks

A payer or payee application may be modeled as shown in Figure 2. The Application Core performs application-specific processing. The WAVE Interface Application Layer (WIAL) processes and transmits information between the Application Core and the WAVE stack. The WIAL and, in some systems, the Payment Service interact with the WAVE stack via the WME SAP and WSM SAP, which are described in IEEE Std 1609.3-2010. Depending on the type of payment system, the Payment Service may provide functions to either the Application Core or the WIAL.

Copyright © 2011 IEEE. All rights reserved.

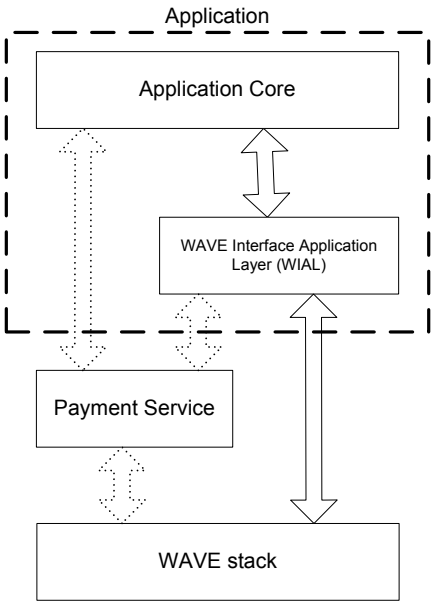

Figure 2—Relationships among application components and Payment Service and WAVE stack

An application transmits and receives payment information either in WAVE short message protocol (WSMP), as defined in IEEE Std 1609.3-2010, or Internet Protocol version 6 (IPv6). Non-payment information may be sent in the same message as payment information.

Electronic payment information may require different processing and handling depending on the type of payment system. For example, after receiving account or identity information from a payer application, a payee application might respond directly with an electronic receipt. In another type of system, the payee might need to send an electronic invoice and receive approval information before it sends the receipt to the payer.

5. Payment Service overview

5.1 Payment Service functions

The functions of a Payment Service can be grouped into the following two categories:

— *General EPS functions:* These may include functions for the origination, communication, processing, storage or retrieval of EPS information. Requirements for these functions are system dependent and are described in the appropriate annex to this standard.

— *Security functions*: These may include functions for encryption and decryption or authentication of EPS information for over-the-air transmission or application-specific purposes. Some systems may use security methods offered by the Payment Service while others may use security methods accessed outside the Payment Service, for example by the application using services described in IEEE P1609.2/D5. Requirements for these functions are system dependent and are described in the appropriate annex to this standard.

Copyright © 2011 IEEE. All rights reserved.

5.2 MIB maintenance

In this version of the standard no management information base (MIB) functions are defined since there are no requirements defined in this revision that require specific MIB attributes for Payment Services.

6. Types of systems using EPS

6.1 Modes of EPS information exchange

6.1.1 Introduction

An EPS system may exchange EPS information between Payment Services in either *Direct* or *Indirect EPS communication* mode.

These modes are discussed in the following subclauses.

6.1.2 Direct EPS communication mode

In a system operating in Direct EPS communication mode, EPS information is directly exchanged between Payment Services via their respective WAVE stacks.

On the sending unit, an application makes a request to its Payment Service for EPS information to be sent. The sending Payment Service then prepares and secures the EPS information before sending it via its WAVE stack. On the receiving unit, the Payment Service extracts the message and sends an appropriate indication to its application.

This mode is illustrated in Figure 3.

Copyright © 2011 IEEE. All rights reserved.

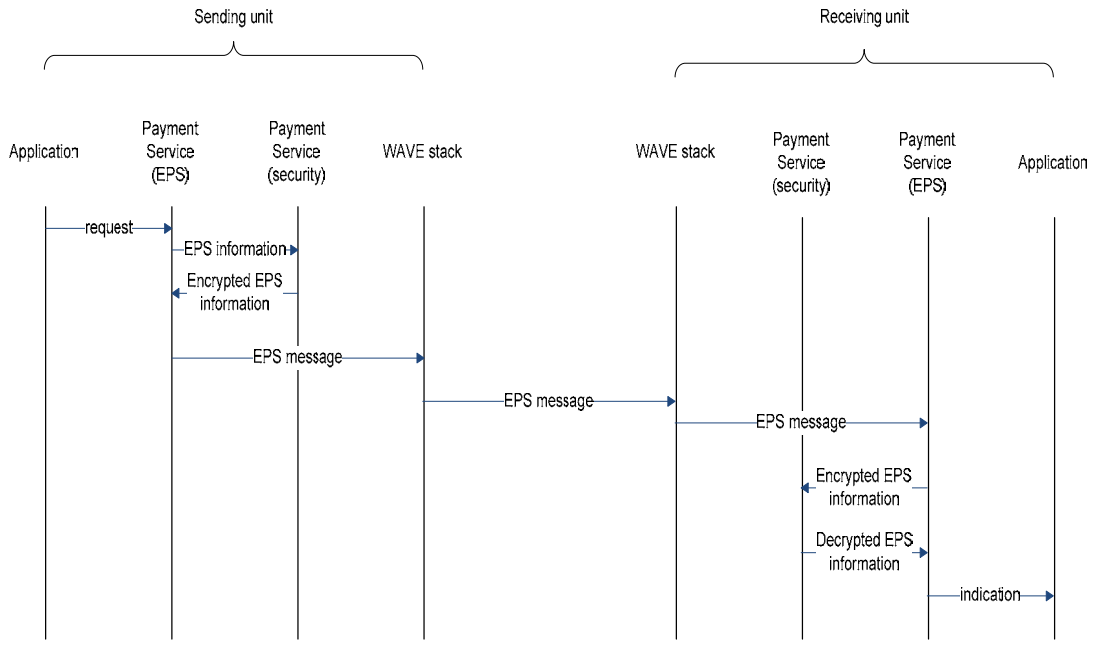

Figure 3—Information flow in Direct EPS communication mode

6.1.3 Indirect EPS communication mode

In a system operating in Indirect EPS communication mode, EPS information is exchanged between Payment Services through their respective applications.

On the sending unit, an application may make a request to its Payment Service for EPS information. The sending Payment Service then prepares the EPS information that is then returned to the application in a confirmation. For example, the preparation of EPS data may involve payment security, such as encryption. The application's WIAL then encapsulates the EPS information and/or a request for EPS information to the receiving unit, along with application-specific information if desired, into a message that is sent over the air via the WAVE stack.

On the receiving unit, the application separates the message into the EPS information and/or requests for EPS information, which is sent to the Payment Service in an indication, and any application information. The Payment Service may then process the EPS information and send a response to the application. The processing of EPS data may involve payment security, such as decryption.

This mode is illustrated in Figure 4.

Copyright © 2011 IEEE. All rights reserved.

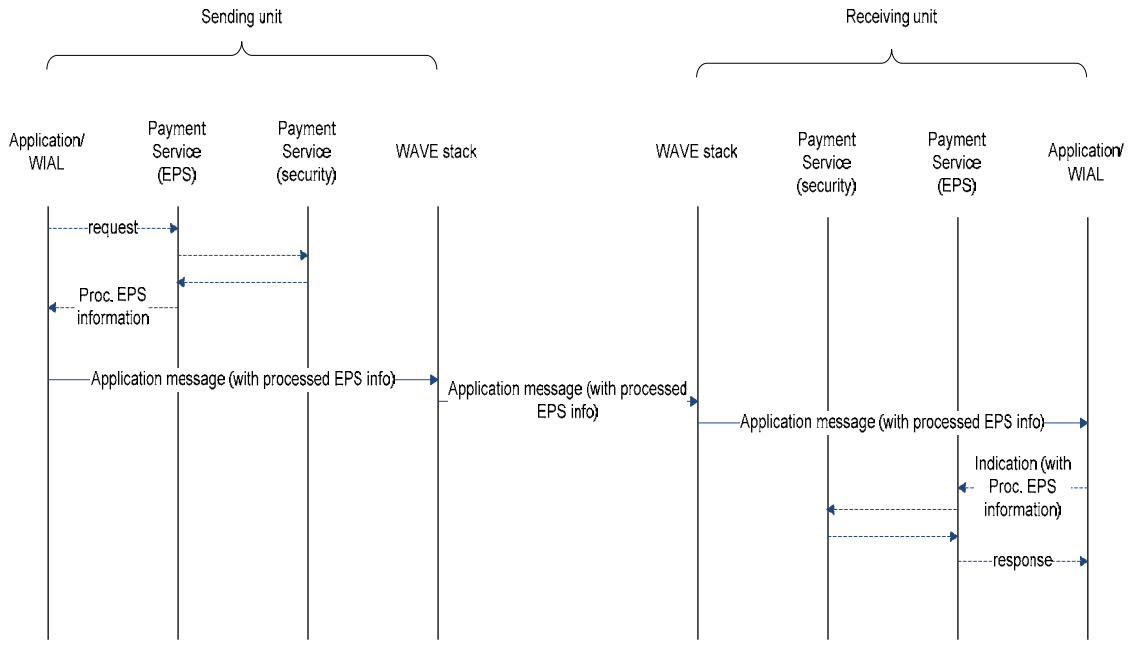

Figure 4—Information flow in Indirect EPS communication mode

6.2 Payer and payee approval types

6.2.1 Introduction

On over-the-air receipt of Payment information and validation/verification, a payer or payee may be required to wait to obtain approval from another entity before it can perform other actions or send a response, or if pre-approved, it would not need to wait.

The types of approval, including examples of situations where they may be required, are defined in 6.2.2 through 6.2.4.

Note that the details of validation/verification and approval, and the circumstances and consequences of disapproval are system-dependent and are out of scope for 6.2.2 through 6.2.4.

6.2.2 Payer response approval types

On over-the-air receipt and validation/verification of a Payment information request from the payee, the payer may require approval (e.g., from a human-machine interface) to reply with a Payment information response. For example, for a financial transaction, the Payment information request may contain an invoice and the Payment information response may contain the account information from which the payment is to be made. For another example, for an identity transaction (i.e., payer authentication without payment), the Payment information request may contain a request for identification and the Payment information response may contain the identity of the payer.

The *Payer response approval types* are defined as follows:

Copyright © 2011 IEEE. All rights reserved.

— *Payer response without pre-approval* means that the payer needs to wait for approval from an external entity before it may send the Payment information response. This approval type is shown in Figure 5.

— *Payer response pre-approval* means that the payer does not need to wait for approval before it may send the Payment information response.

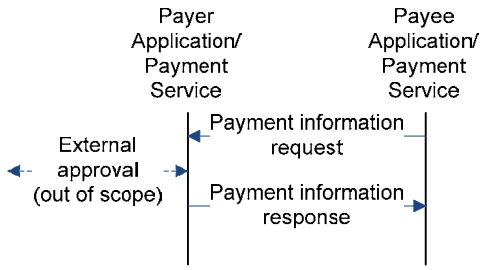

Figure 5—Over-the-air payment information handling (payer response without pre-approval)

6.2.3 Payee action approval types

On over-the-air receipt and validation/verification of a Payment information response from the payer as described in 6.2.2, the payee may require approval (e.g., from a central service) to take further action. Further action, for example, may include sending to the payer a Payment information confirmation that may include the receipt for payment for a financial transaction or the approval of identity for an identity transaction.

The *Payee action approval types* are as follows:

— *Payee action without pre-approval* means that the payee needs to wait for approval from an external entity before it may take further action. This approval type is shown in Figure 6.

— *Payee action pre-approval* means that the payee does not need to wait for approval before it takes further action.

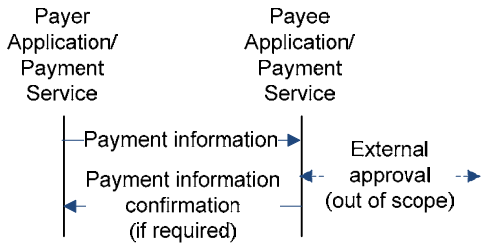

Figure 6—Over-the-air EPS information handling (payee action without pre-approval)

6.2.4 Payer action approval types

On sending a Payment information response to the payee as described in 6.2.2, the payer may require approval from the payee (i.e., acceptable Payment information confirmation as described in 6.2.3) to take

Copyright © 2011 IEEE. All rights reserved.

further action. For example, further action for an identity transaction may include acting on approval to access a facility.

The *Payer action approval types* are as follows:

— *Payer action without approval* means that the payer must wait until it receives the Payment information confirmation from the payee before taking further action.

— *Payer action pre-approval* means that the payer does not need to wait for approval before taking further action.

Copyright © 2011 IEEE. All rights reserved.

Annex A

(normative)

Profile for Electronic Fee Collection (EFC) systems per ISO 14906 and ISO 15628

A.1 Introduction

An EFC system is a special case of an EPS system.

For one type of EFC system, the relationship between an application and its WAVE Interface Application Layer (WIAL) is defined in ISO 14906 and ISO 15628. This annex defines additional requirements for such an application to interact with a WAVE stack. Hereafter in this annex, all references to EFC refer to instances of this type of EFC system.

An EFC system functions in Payment Service via Indirect EPS communication mode as described in 6.1.3.

ISO 14906 specifies an application interface for EFC systems that uses a DSRC medium. More specifically, it is a "toolbox" for defining EFC-application transactions that provides specifications for the EFC transaction model, EFC data elements (referred to as attributes) and functions from which specific EFC transactions can be built. It supports and enables interoperability between EFC systems on an application interface level, but does not propose a specific solution for interoperability on the over-the-air link.

The interface definition itself is independent of a specific medium and can therefore be used on top of any lower layers, for example the WAVE stack, by using a common application layer. ISO 15628 defines this needed application layer by providing appropriate communication tools in the form of services primitives for applications. It assumes a generic service interface including functional services to the lower layers and their functional requirements.

A.2 General description

On an EFC payment equipment unit, a Payment Service offers security and payment functions to its Application Core.

In an EFC system, payer equipment units, which host payer applications, are on-board units (OBU) and payee equipment units, which host payee applications, are on road-side units (RSU).

The scope of this annex, shown in Figure A.1, encompasses the payer and payee Payment Services, WAVE stacks and applications' WIALs and the interfaces among them. The Application Cores are out of scope, but the interfaces to them are in scope.

Copyright © 2011 IEEE. All rights reserved.

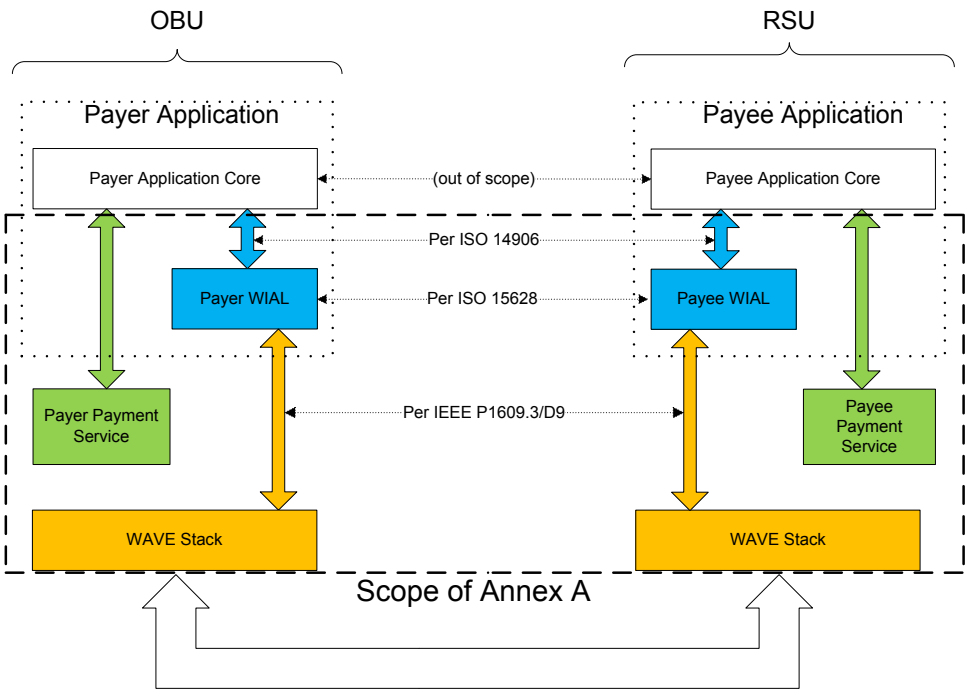

Figure A.1—Scope of Annex A

The WIAL-to-Application Core interface shall be as specified in ISO 14906 and ISO 15628.

The WIAL-to-WAVE stack interface shall be as specified in IEEE Std 1609.3-2010.

Security functions such as encryption and calculation of Message Authentication Codes shall use the 128-bit Advanced Encryption Standard (AES-128), as described in FIPS Publication 197 (November 2001).

The Payment Service and its interfaces are specified in the body of the standard and in this annex.

In an EFC application, the Application Core and WAVE Interface Application Layer (WIAL) correspond to the Application Process and DSRC Application Layer respectively as defined in ISO 14906.

The contents of the Beacon Service Table (BST), defined in ISO 14906, along with optional application-specific information, shall be transmitted as the Provider Service Context (PSC) of a WAVE service advertisement (WSA) message, as defined in IEEE Std 1609.3-2010. The contents of the PSC shall be as follows:

```
ProviderServiceContext ::= SEQUENCE {
     bst                      BST,-- see ISO 14906
     applicationSpecificInfo  OCTET STRING (SIZE(0..18))
}
```

All other Transfer-Application Data Units (T-ASDUs), as defined in ISO 14906, shall be carried in the payloads of data frames sent using WAVE short message protocol (WSMP) as defined in IEEE Std 1609.3-2010. The Logical Link Control Identifier (LID) defined in ISO 15628 shall be replaced by the appropriate MAC address in the WAVE stack.

Copyright © 2011 IEEE. All rights reserved.

A.3 WIAL and WAVE services

A.3.1 Introduction

The correspondence between WIAL and WAVE functions is shown in Table A.1, and will be discussed in the following subclauses.

Table A.1—WIAL and WAVE functions

WIAL function (ISO 15628)	EFC function (ISO 14906)	WAVE primitive(s) (IEEE Std 1609.3-2010)
INITIALISATION (see Note below)	N/A	WME-ProviderService.request, WME-UserService.request
GET	N/A	WSM-WaveShortMessage.request, WSM-WaveShortMessage.indication
SET	N/A	
ACTION	GET_STAMPED	
ACTION	GET_INSTANCE	
ACTION	SET_MMI	
ACTION	ECHO	
ACTION	GET_SECURE	
ACTION	SET_SECURE	
EVENT-REPORT	RELEASE	WME-ProviderService.request

NOTE—The spelling of "initialisation" is as given in ISO 14906.

GET_SECURE and SET_SECURE shall not use access credentials.

A.3.2 GET_SECURE

The coding of the data types for the action and response parameters of the GET_SECURE function defined in ISO 14906 are given in more detail as follows:

```
Authenticator ::= OCTET STRING

AttributeListEncrypted::= OCTET STRING

GETSECURE-Request::= Action-Request (WITH COMPONENTS {..., actionType (2),
actionParameter (GetSecureRq)})

GETSECURE-Response::= Action-Response (WITH COMPONENTS {..., responseParameter
(GetSecureRs)})

GetSecureRq::= OCTET STRING (CONTAINING ContentOfGetSecureRq)

GetSecureRs ::= OCTET STRING (CONTAINING ContentOfGetSecureRs)

ContentOfGetSecureRq ::= SEQUENCE {
             initialValue               OCTET STRING,
             keyRefEncryption           INTEGER (0..255),
             rndPayee                   OCTET STRING,
             requests     SEQUENCE OF (SIZE(0..127,...)) GSRequests
             }
```


Copyright © 2011 IEEE. All rights reserved.

```
GSRequests::= SEQUENCE {
             fill                       BIT STRING (SIZE(5)), --fill bits
             gSRequest                  GSRequest
             }

GSRequest ::= CHOICE{
             choice1                    GSRq-NoMAC-NoInstances,
             choice2                    GSRq-MAC-NoInstances,
             choice3                    GSRq-NoMAC-Instances,
             choice4                    GSRq-MAC-Instances,
             rfu1                       NULL -- reserved for future use
             }

GSRq-NoMAC-NoInstances::= AttributeIdList

GSRq-MAC-NoInstances::= SEQUENCE{
             attributeIdList            AttributeIdList,
             keyRefMac                  INTEGER(0..255)
             }

GSRq-NoMAC-Instances::= SEQUENCE{
             posOfFirstInstance         INTEGER(0..255),
             posOfLastInstance          INTEGER(0..255),
             attributeIdList            AttributeIdList
             }

GSRq-MAC-Instances::= SEQUENCE{
             posOfFirstInstance         INTEGER(0..255),
             posOfLastInstance          INTEGER(0..255),
             attributeIdList            AttributeIdList,
             keyRefMac                  INTEGER(0..255)
             }


ContentOfGetSecureRs::= SEQUENCE (SIZE(0..127,...)) OF GSResponses

GSResponses::= SEQUENCE {
             fill                       BIT STRING (SIZE(5)),
             gSResponse                 GSResponse
             }

GSResponse::= CHOICE {
             choice1                    GSRs-NoMAC-NoInstances,
             choice2                    GSRs-MAC-NoInstances,
             choice3                    GSRs-NoMAC-Instances,
             choice4                    GSRs-MAC-Instances,
             rfu2                       NULL
             }


GSRs-NoMAC-NoInstances::= AttributeListEncrypted

GSRs-MAC-NoInstances::= SEQUENCE{
             attributeListEncrypted     AttributeListEncrypted,
             authenticator              Authenticator
             }

GSRs-NoMAC-Instances::= AttributeListEncrypted

GSRs-MAC-Instances::= SEQUENCE{
             attributeListEncrypted     AttributeListEncrypted,
             authenticator              Authenticator
             }
```


Copyright © 2011 IEEE. All rights reserved.

The data semantics are given in the following table:

Data field	Range	Semantics
attributeIdList	—	List of IDs of the requested attributes.
posOfFirstInstance	0–255	If present, index of first instance to be read-out, see GET_INSTANCE. If not present, only the first instance is read-out, i.e., similar to GET.
posOfLastInstance	0–255	Only present if previous data element is present; index of last instance to be read-out, see GET_INSTANCE.
initialValue	16 Octets	Initial value used for encryption.
keyRefEncryption	0–255	Reference to the derived encryption key in the OBU (i.e., its attribute ID).
rndPayee	16 Octets	Random number—challenge for MAC calculation.
keyRefMac	0–255	Reference to the derived authentication key in the OBU (i.e., its attribute ID). If not present no authenticator is required in the response.
attributeListEncrypted	—	Encrypted application data in the form of an encrypted AttributeList.
authenticator	16 Octets	MAC calculated by the recipient over the requested data using rndPayee and keyRefMAC.

The attributeListEncrypted that is returned in the GET_SECURE.response is the result of the encryption of the following:

— The attributeList corresponding to the attributeIdList, if no instances are addressed.

— An attributeList that contains for each attribute a series of attribute values corresponding to the instances from posOfFirstInstance to posOfLastInstance.

A.3.3 SET_SECURE

The coding of the data types for the action and response parameters of the SET_SECURE function defined in ISO 14906 are given in more detail as follows:

```
SETSECURE-Request::= Action-Request (WITH COMPONENTS {..., actionType (3),
actionParameter (SetSecureRq)})

SETSECURE-Response::= Action-Response (WITH COMPONENTS {..., responseParameter
(SetSecureRs)})

SetSecureRq::= OCTET STRING (CONTAINING ContentOfSetSecureRq)

SetSecureRs ::= OCTET STRING (CONTAINING ContentOfSetSecureRs)

ContentOfSetSecureRq ::= SEQUENCE {
             initialValue               OCTET STRING,
             keyRefEncryption           INTEGER (0..255),
             RndPayee                   OCTET STRING,
             requests     SEQUENCE (SIZE(0..127,...)) OF SSRequests
             }
```


Copyright © 2011 IEEE. All rights reserved.

```
SSRequests::= SEQUENCE {
             fill                        BIT STRING (SIZE(5)),
             sSRequest                   SSRequest
             }

SSRequest ::= CHOICE{
             choice1                     SSRq-NoMacRs,
             choice2                     SSRq-MacRs,
             choice3                     SSRq-NoMacRs-Instances,
             rfu3                        NULL,
             rfu4                        NULL
             }


SSRq-NoMacRs::= SEQUENCE {
             attributeListEncrypted      AttributeListEncrypted,
             keyRefMacRq                 INTEGER(0..255),
             reqAuthenticator            Authenticator
             }


SSRq-MacRs::= SEQUENCE {
             attributeListEncrypted      AttributeListEncrypted,
             keyRefMacRq                 INTEGER(0..255),
             reqAuthenticator            Authenticator,
             keyRefMacRs                 INTEGER(0..255)
             }

SSRq-NoMacRs-Instances::= SEQUENCE {
             posOfInstance               INTEGER(0..255),
             attributeListEncrypted      AttributeListEncrypted,
             keyRefMacRq                 INTEGER(0..255),
             reqAuthenticator            Authenticator
             }

ContentOfSetSecureRs::= SEQUENCE (SIZE(0..127,...)) OF SSResponses

SSResponses::= SEQUENCE {
             fill                        BIT STRING (SIZE(5)),
             sSResponse                  SSResponse
             }

SSResponse::= CHOICE {
             choice1       SSRs-NoMacRs,
             choice2       SSRs-MACRs,
             rfu5          NULL,
             rfu6          NULL,
             rfu7          NULL
             }

SSRs-NoMacRs::= NULL
SSRs-MACRs::= Authenticator
```


Copyright © 2011 IEEE. All rights reserved.

The data semantics are given in the following table:

Data field	Range	Semantics
attributeListEncrypted	—	Encrypted application data, in the form of an encrypted AttributeList.
posOfInstance	0–255	If present, index of the instance to be written.
initialValue	16 Octets	Initial value used for encryption.
keyRefEncryption	0–255	Reference to the derived encryption key in the OBU (i.e., its attribute ID).
RndPayee	16 Octets	Random number—challenge for MAC calculation.
reqAuthenticator	16 Octets	Authenticator to be checked by the OBU before setting the attribute's value using rndPayer and keyRefMACReq.
keyRefMacReq	0–255	Reference to the derived authentication key in the OBU (i.e., its attribute ID) for checking the reqAuthenticator.
keyRefMacResp	0–255	Reference to the derived authentication key in the OBU (i.e., its attribute ID) for calculating the Authenticator.
authenticator	16 Octets	MAC calculated by the OBU over the requested data using RndPayee and KeyRefMACResp.

The attributeListEncrypted that is sent in the SET_SECURE.request is the result of the encryption of the following:

— The attributeList, if no instances are addressed

— An attributeList that contains for each attribute the value of the instance corresponding to posOfInstance.

A.3.4 Payee WIAL

Upon acceptance of RegisterApplicationRSU.request (as defined in ISO 14906) from the Payee Application Core, the payee WIAL shall request WAVE provider services by submitting the WME-ProviderService.request (as defined in IEEE Std 1609.3-2010) with parameters as required by the application.

The WAVE stack will then broadcast Vendor Specific Action frames, as defined in IEEE Std 802.11, containing WSAs allowing EFC and application-specific transactions to occur.

Upon acceptance of DeregisterApplicationRSU.request (as defined in ISO 14906) from the Payee Application Core, the payee WIAL shall delete the WAVE provider service by submitting the WME-ProviderService.request (as defined in IEEE Std 1609.3-2010).

The WAVE stack will then stop broadcasting WSAs, prohibiting subsequent EFC and application-specific transactions.

An example of the payee WIAL managing WAVE services is shown in Figure A.2.

Copyright © 2011 IEEE. All rights reserved.

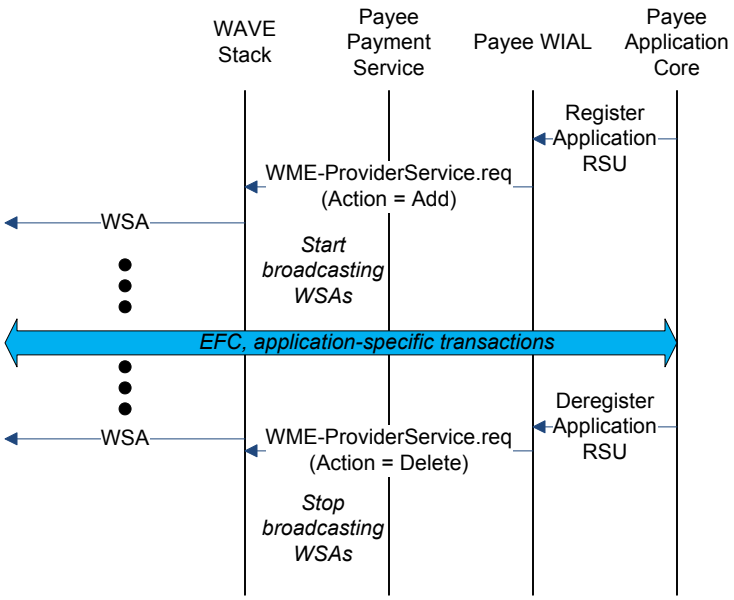

Figure A.2—Example of EFC Payee and WAVE provider services

A.3.5 Payer WIAL

Upon acceptance of RegisterApplicationOBU.request (as defined in ISO 14906) from the payer Application Core, the payer WIAL shall request WAVE user services by submitting the WME-UserService.request (as defined in IEEE Std 1609.3-2010) with parameters as required by the application.

The WAVE stack will then listen for WSA messages.

Upon acceptance of DeregisterApplicationOBU.request (as defined in ISO 14906) from the Payer Application Core, the payer WIAL shall delete the WAVE user service by submitting the WME-UserService.request (as defined in IEEE Std 1609.3-2010).

The WAVE stack will then stop listening for WSA messages.

An example of the payer WIAL managing WAVE services is shown in Figure A.3.

Copyright © 2011 IEEE. All rights reserved.

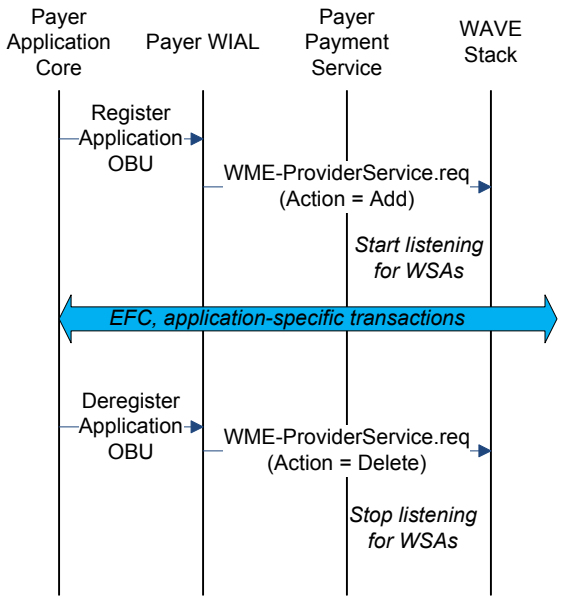

Figure A.3—Example of EFC Payer and WAVE user services

A.4 Interfaces

A.4.1 Overview

The following subclauses describe the primitives used on the service access points (SAPs) on the payer and payee WIALs and Payment Services.

Copyright © 2011 IEEE. All rights reserved.

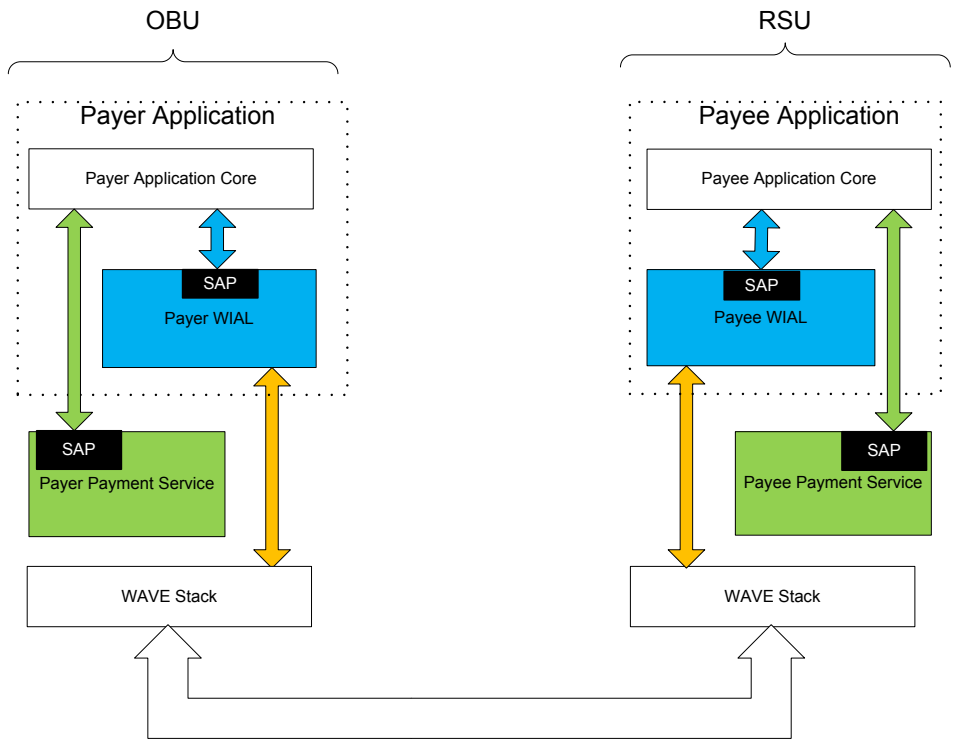

Figure A.4—Service access points

The primitives are listed in Table A.2.

Copyright © 2011 IEEE. All rights reserved.

Table A.2—List of primitives

SAP	Primitive	Specified in
Payer Payment Service SAP	PS-Encrypt.request	A.4.2.2
	PS-Encrypt.confirm	A.4.2.3
	PS-Decrypt.request	A.4.2.4
	PS-Decrypt.confirm	A.4.2.5
	PS-CalculateMessageAuthenticationCode.request	A.4.2.6
	PS-CalculateMessageAuthenticationCode.confirm	A.4.2.7
	PS-StoreKey.request	A.4.2.8
	PS-StoreKey.confirm	A.4.2.9
	PAYERPS-GetPaymentData.request	A.4.3.1
	PAYERPS-GetPaymentData.confirm	A.4.3.2
	PAYERPS-SaveReceipts.request	A.4.3.3
	PAYERPS-SaveReceipts.confirm	A.4.3.4
	PS-Encrypt.request	A.4.2.2
	PS-Encrypt.confirm	A.4.2.3
	PS-RetrieveKeyMaterial.request	A.4.2.10
	PS-RetrieveKeyMaterial.confirm	A.4.2.11
Payee Payment Service SAP	PS-Encrypt.request	A.4.2.2
	PS-Encrypt.confirm	A.4.2.3
	PS-Decrypt.request	A.4.2.4
	PS-Decrypt.confirm	A.4.2.5
	PS-CalculateMessageAuthenticationCode.request	A.4.2.6
	PS-CalculateMessageAuthenticationCode.confirm	A.4.2.7
	PS-GenerateKeyMaterial.request	A.4.2.12
	PS-GenerateKeyMaterial.confirm	A.4.2.13
	PAYEEPS-MakePayment.request	A.4.4.1
	PAYEEPS-MakePayment.confirm	A.4.4.2

A.4.2 General Payment Service primitives

A.4.2.1 Overview

The payer and payee Payment Service entities shall each provide the following functions for its corresponding Application Core.

A.4.2.2 PS-Encrypt.request

A.4.2.2.1 Function

This primitive encrypts an octet string per the requirements in Clause 5.

A.4.2.2.2 Semantics of the service primitive

The parameters of the PS-Encrypt.request primitive are as follows:

```
PS-Encrypt.request(
        Data,
        InitialValue,
```


Copyright © 2011 IEEE. All rights reserved.

DiversifierData,

EFC-ContextMark,

KeyRef

)

Name	Type	Valid range	Description
Data	AttributeList	As defined in ISO 14906	Data to be encrypted
InitialValue	Octet string	Octet string of length 16	Initial value for encryption
DiversifierData	Octet string	Octet string of length 16	Used to reference or derive encryption key
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	Used to reference encryption key material
KeyRef	INTEGER	0–255	(optional) Reference to the encryption key
KeyInstance	INTEGER	0–255	(optional) Reference to the key instance

A.4.2.2.3 When generated

This primitive is generated by an Application Core when it wishes to encrypt an octet string.

A different value of InitialValue should be used every time this primitive is called. This InitialValue need not be unpredictable.

A.4.2.2.4 Effect of receipt

The Payment Service shall perform the encryption and issue a PS-Encrypt.confirm that reflects the result. The encryption shall be performed using an approved algorithm as listed in A.6.1.1 of this document.

The Payment Service shall obtain the encryption key in one of the following two ways.

If the (KeyRef, KeyInstance, EFC-ContextMark) have already been stored within the Payment Service by a call to PS-StoreKey, the Payment Service shall use the provided KeyRef, KeyInstance, and EFC-ContextMark parameters to obtain the key.

Otherwise, the Payment Service shall use the EFC-ContextMark and KeyRef to look up an Encryption Master Key. It shall then use the Encryption Master Key and the DiversifierData as the Key and Data inputs, respectively, to an approved Key Derivation algorithm as defined in A.6.1.3. The output from this key derivation process shall be the encryption key.

NOTE—In general the Payer will use the first method (recovering a stored key) and the Payee will use the second method (deriving the key from the DiversifierData).

A.4.2.3 PS-Encrypt.confirm

A.4.2.3.1 Function

This primitive reports the results of an encryption.

Copyright © 2011 IEEE. All rights reserved.

A.4.2.3.2 Semantics of the service primitive

The parameters of the PS-Encrypt.confirm primitive are as follows:

PS-Encrypt.confirm(
ResultCode
EncryptedData,
)

Name	Type	Valid range	Description
ResultCode	Enumerated	Success, Unable to obtain key	Result of request operation
EncryptedData	Octet string	N/A	(optional) Encrypted data, included if ResultCode is success

A.4.2.3.3 When generated

This primitive is generated by a Payment Service as a result of PS-Encrypt.request.

A.4.2.3.4 Effect of receipt

The corresponding Application Core is notified of the results of the encryption.

A.4.2.4 PS-Decrypt.request

A.4.2.4.1 Function

This primitive decrypts an octet string per the requirements in Clause 5.

A.4.2.4.2 Semantics of the service primitive

The parameters of the PS-Decrypt.request primitive are as follows:

PS-Decrypt.request(
Data,
InitialValue,
DiversifierData,
EFC-ContextMark,
KeyRef
)

Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
Data	Octet string	N/A	Data to be decrypted
InitialValue	Octet string	16 octets	Initial value for decryption
DiversifierData	Octet string	Octet string of length 16	Used to reference or derive decryption key
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	Used to reference decryption key material
KeyRef	INTEGER	0–255	(optional) Reference to the master encryption key
KeyInstance	INTEGER	0–255	(optional) Reference to the key instance

A.4.2.4.3 When generated

This primitive is generated by an Application Core when it wishes to decrypt an octet string.

A.4.2.4.4 Effect of receipt

The Payment Service shall perform the decryption and issue a PS-Decrypt.confirm that reflects the result.

The decryption shall be performed using an approved algorithm as listed in A.6.1.1.

The use of InitialValue is described in A.6.1.1.

The Payment Service shall obtain the decryption key in one of the following two ways.

If the (KeyRef, KeyInstance, EFC-ContextMark) have already been stored within the Payment Service by a call to PS-StoreKey, the Payment Service shall use the provided KeyRef, KeyInstance, and EFC-ContextMark parameters to obtain the key.

Otherwise, the Payment Service shall use the EFC-ContextMark and KeyRef to look up an Encryption Master Key. It shall then use the Encryption Master Key and the DiversifierData as the Key and Data inputs, respectively, to an approved Key Derivation algorithm as defined in A.6.1.3. The output from this key derivation process shall be the decryption key.

NOTE—In general the Payer will use the first method (recovering a stored key) and the Payee will use the second method (deriving the key from the DiversifierData).

A.4.2.5 PS-Decrypt.confirm

A.4.2.5.1 Function

This primitive reports the results of a decryption.

A.4.2.5.2 Semantics of the service primitive

The parameters of the PS-Decrypt.confirm primitive are as follows:

PS-Decrypt.confirm(

Copyright © 2011 IEEE. All rights reserved.

ResultCode,

DecryptedData,

)

Name	Type	Valid range	Description
ResultCode	Enumerated	Success, Unpadding failure, could not obtain key	Result
DecryptedData	Octet String	Any octet String	(optional) Decrypted data. Only included if ResultCode is "Success"

A.4.2.5.3 When generated

This primitive is generated by a Payment Service as a result of PS-Decrypt.request.

A.4.2.5.4 Effect of receipt

The corresponding Application Core is notified of the results of the decryption.

A.4.2.6 PS-CalculateMessageAuthenticationCode.request

A.4.2.6.1 Function

This primitive calculates a Message Authentication Code.

A.4.2.6.2 Semantics of the service primitive

The parameters of the PS-CalculateMessageAuthenticationCode.request primitive are as follows:

PS-CalculateMessageAuthenticationCode.request(

Data,

Rnd,

EFC-ContextMark,

DiversifierData,

KeyRefMac

)

Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
Data	Octet String	An encoded Attribute List	Data on which the Message Authentication Code is to be calculated
Rnd	Octet string	String of length 16	Random number used for calculation of the Message Authentication Code
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	Used to reference or derive authentication key
DiversifierData	Octet String	String of length 16	Used to reference or derive authentication key
KeyRefMac	INTEGER	0–255	Reference to the authentication key

A.4.2.6.3 When generated

This primitive is generated by an Application Core when it wishes to calculate a Message Authentication Code.

A.4.2.6.4 Effect of receipt

The Payment Service shall perform the calculation and issue a PS-CalculateMessageAuthenticationCode.confirm that reflects the result. The Message Authentication Code shall be calculated using an approved algorithm as specified in A.6.1.2. The input message M shall be formed by concatenating the Data parameter and the Rnd parameter.

The Payment Service shall obtain the authentication key in one of the following two ways.

If the (KeyRef, DiversifierData, EFC-ContextMark) have already been stored within the Payment Service by a call to PS-StoreKey, the Payment Service shall use the provided KeyRef, DiversifierData, and EFC-ContextMark parameters to obtain the key.

Otherwise, the Payment Service shall use the EFC-ContextMark and KeyRef to look up an Authentication Master Key. It shall then use the Authentication Master Key and the DiversifierData as the Key and Data inputs, respectively, to an approved Key Derivation algorithm as defined in A.6.1.3. The output from this key derivation process shall be the decryption key.

NOTE—In general the Payer will use the first method (recovering a stored key) and the Payee will use the second method (deriving the key from the DiversifierData).

A.4.2.7 PS-CalculateMessageAuthenticationCode.confirm

A.4.2.7.1 Function

This primitive reports the Message Authentication Code.

A.4.2.7.2 Semantics of the service primitive

The parameters of the PS-CalculateMessageAuthenticationCode.confirm primitive are as follows:

Copyright © 2011 IEEE. All rights reserved.

PS- CalculateMessageAuthenticationCode.confirm(

MessageAuthenticationCode,

)

Name	Type	Valid range	Description
MessageAuthenticationCode	Octet string	String of length 16	Message Authentication Code

A.4.2.7.3 When generated

This primitive is generated by a Payment Service as a result of PS-CalculateMessageAuthenticationCode.request.

A.4.2.7.4 Effect of receipt

The corresponding Application Core is notified of the Message Authentication Code.

A.4.2.8 PS-StoreKey.request

A.4.2.8.1 Function

This primitive stores a key.

A.4.2.8.2 Semantics of the service primitive

The parameters of the PS-StoreKey.request primitive are as follows:

PS-StoreKey.request(

EFC-ContextMark

Diversifier

KeyData,

KeyType,

KeyRef,

KeyInstance

)

Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
Diversifier	Octet String	String of length 16	The Diversifier associated with the key
KeyData	Octet String	String of length 16 or 32	The key data
KeyType	Enumerated	Encryption, Authentication	Type of key
KeyRef	INTEGER	0–255	Optional. Value to be used to refer to the key
KeyInstance	INTEGER	0–255	Optional. Value to be used to refer to the key

A.4.2.8.3 When generated

This primitive is generated by an Application Core when it wishes to store a key for use in encryption, or authentication.

A.4.2.8.4 Effect of receipt

The Payment Service shall store the key, associated with the given KeyRef and Diversifier.

A.4.2.9 PS-StoreKey.confirm

A.4.2.9.1 Function

This primitive reports the result of the corresponding request.

A.4.2.9.2 Semantics of the service primitive

The parameters of the PS-StoreKey.confirm primitive are as follows:

```
PS-StoreKey.confirm(
        ResultCode,
        )
```

Name	Type	Valid range	Description
ResultCode	Enumerated	Success, failure	Result of corresponding request.

A.4.2.9.3 When generated

This primitive is generated by a Payment Service as a result of PS-StoreKey.request.

Copyright © 2011 IEEE. All rights reserved.

A.4.2.9.4 Effect of receipt

The corresponding Application Core is notified whether the key has been successfully stored.

A.4.2.10 PS-RetrieveKeyMaterial.request

A.4.2.10.1 Function

This primitive requests the information needed to refer to a key.

A.4.2.10.2 Semantics of the service primitive

The parameters of the PS-RetrieveKeyMaterial.request primitive are as follows:

```
PS-RetrieveKeyMaterial.request(
        EFC-ContextMark
        KeyType,
        KeyRef,
        KeyInstance
        )
```

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
KeyType	Enumerated	Encryption, Authentication	Type of key
KeyRef	INTEGER	0–255	Optional. Value to be used to refer to the key
KeyInstance	INTEGER	0–255	Optional. Value to be used to refer to the key

A.4.2.10.3 When generated

This primitive is generated by a Payer Application Core when it wishes to retrieve a key for use in encryption, or authentication.

A.4.2.10.4 Effect of receipt

The Payment Service shall attempt to retrieve the key from storage.

A.4.2.11 PS-RetrieveKeyMaterial.confirm

A.4.2.11.1 Function

This primitive reports the result of the corresponding request.

Copyright © 2011 IEEE. All rights reserved.

A.4.2.11.2 Semantics of the service primitive

The parameters of the PS-RetrieveKeyMaterial.confirm primitive are as follows:

PS-RetrieveKeyMaterial.confirm(
ResultCode,
Diversifier,
KeyData
)

Name	Type	Valid range	Description
ResultCode	Enumerated	Success, failure	Result of corresponding request.
Diversifier	Octet String	String of length 16	If ResultCode is success, The Diversifier associated with the key
KeyData	AttributeListEncrypted	String of length 16 or 32	If ResultCode is success, the key data

A.4.2.11.3 When generated

This primitive is generated by a Payment Service as a result of PS-RetrieveKeyMaterial.request.

A.4.2.11.4 Effect of receipt

The corresponding Application Core has access to the Diversifier and can use it within a communications session.

A.4.2.12 PS-GenerateKeyMaterial.request

A.4.2.12.1 Function

This primitive generates a key and Diversifier.

A.4.2.12.2 Semantics of the service primitive

The parameters of the PS-GenerateKeyMaterial.request primitive are as follows:

PS-GenerateKeyMaterial.request(
EFC-ContextMark
KeyType,
KeyRef,
KeyInstance
)

Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
KeyType	Enumerated	Encryption, Authentication	Type of key
KeyRef	INTEGER	0–255	Value to be used to refer to the key
KeyInstance	INTEGER	0–255	Value to be used to refer to the key

A.4.2.12.3 When generated

This primitive is generated by a Payee Application Core when it wishes to generate an encryption key and Diversifier to be installed on a Payer unit.

A.4.2.12.4 Effect of receipt

The Payment Service shall attempt to generate a Diversifier and the associated key.

A.4.2.13 PS-GenerateKeyMaterial.confirm

A.4.2.13.1 Function

This primitive reports the result of the corresponding request.

A.4.2.13.2 Semantics of the service primitive

The parameters of the PS-GenerateKeyMaterial.confirm primitive are as follows:

```
PS-GenerateKeyMaterial.confirm(
        ResultCode,
        Diversifier,
        KeyData
        )
```

Name	Type	Valid range	Description
ResultCode	Enumerated	Success, failure	Result of corresponding request.
Diversifier	Octet String	String of length 16	If ResultCode is success, The Diversifier associated with the key
KeyData	AttributeListEncrypted	String of length 16 or 32	If ResultCode is success, the key data

Copyright © 2011 IEEE. All rights reserved.

A.4.2.13.3 When generated

This primitive is generated by a Payment Service as a result of PS-GenerateKeyMaterial.request.

A.4.2.13.4 Effect of receipt

The corresponding Application Core is notified whether the key has been successfully generated.

A.4.3 Payer Payment Service primitives

A.4.3.1 PAYERPS-GetPaymentData.request

A.4.3.1.1 Function

This primitive requests that the payer Payment Service return the PaymentMeans and the last and penultimate Receipt data, as defined in ISO 14906.

A.4.3.1.2 Semantics of the service primitive

The parameters of the PAYERPS-GetPaymentData.request primitive are as follows:

PAYERPS-GetPaymentData.request(

EFC-ContextMark,

)

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated

A.4.3.1.3 When generated

This primitive is generated by a payer Application Core when it wishes to retrieve the PaymentMeans and the last and penultimate Receipt data.

A.4.3.1.4 Effect of receipt

The Payment Service shall issue a PAYERPS-GetPaymentData.confirm that reflects the result.

A.4.3.2 PAYERPS-GetPaymentData.confirm

A.4.3.2.1 Function

This primitive reports the PaymentMeans and last and penultimate Receipt data.

Copyright © 2011 IEEE. All rights reserved.

A.4.3.2.2 Semantics of the service primitive

The parameters of the PAYERPS-GetPaymentData.confirm primitive are as follows:

PAYERPS-GetPaymentData.confirm(
EFC-ContextMark,
ResultCode,
PaymentMeansData,
Receipt,
Receipt1,
Receipt2
)

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
ResultCode	Enumeration	SUCCESS, NOT_FOUND	Indicates the result of PAYERPS-GetPaymentData.request
PaymentMeansData	AttributeList / AttributeListEncrypted	As defined in ISO 14906	PaymentMeans data.
Receipt	AttributeList	As defined in ISO 14906	Optional. List of attributes containing receipt information
Receipt1	Receipt	As defined in ISO 14906	Receipt information
Receipt2	Receipt	As defined in ISO 14906	Receipt information

Any combination of attributes in the data group RECEIPT, as defined in ISO 14906, may be used as parameters for Receipt. An Application Core shall support ReceiptData1 and ReceipData2.

A.4.3.2.3 When generated

This primitive is generated by a payer Payment Service as a result of PAYERPS-GetPaymentData.request.

A.4.3.2.4 Effect of receipt

The corresponding payer Application Core is notified of the results of the payer Payment Service's data retrieval.

A.4.3.3 PAYERPS-SaveReceipts.request

A.4.3.3.1 Function

This primitive requests that the payer Payment Service store the Receipt data for the current and previous transaction, as defined in ISO 14906.

Copyright © 2011 IEEE. All rights reserved.

A.4.3.3.2 Semantics of the service primitive

The parameters of the PAYERPS-SaveReceipts.request primitive are as follows:

PAYERPS-SaveReceipts.request(
EFC-ContextMark,
Receipt,
Receipt1,
Receipt2
)

Name	Type	Valid range	Description
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
Receipt	AttributeList	As defined in ISO 14906	Optional. List of attributes containing receipt information
Receipt1	ReceiptData2	As defined in ISO 14906	Receipt information
Receipt2	ReceiptData2	As defined in ISO 14906	Receipt information

Any combination of attributes in the data group RECEIPT, as defined in ISO 14906, may be used as parameters for Receipt.

A.4.3.3.3 When generated

This primitive is generated by a payer Application Core when it wishes to store the Receipt data for the current and previous transaction.

A.4.3.3.4 Effect of receipt

The Payment Service shall issue a PAYERPS-SaveReceipts.confirm that reflects the result.

A.4.3.4 PAYERPS-SaveReceipts.confirm

A.4.3.4.1 Function

This primitive reports the results of the PAYERPS-SaveReceipts.request primitive.

A.4.3.4.2 Semantics of the service primitive

The parameters of the PAYERPS-SaveReceipts.confirm primitive are as follows:

PAYERPS-SaveReceipts.confirm(
    EFC-ContextMark,
    ResultCode
)

Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
AID	DSRCApplicationEntityID	As defined in ISO 15628	Identifies the type of application. For EFC AID= 1
EFC-ContextMark	Octet String	Valid encoding of EFC-ContextMark as defined in ISO 14906	The EFC-ContextMark with which the key is associated
ResultCode	Enumeration	SUCCESS, SAVE_FAILED	Indicates the result of PAYERPS-SubmitEPSData.request

A.4.3.4.3 When generated

This primitive is generated by a payer Payment Service as a result of PAYERPS-SaveReceipts.request.

A.4.3.4.4 Effect of receipt

The corresponding payer Application Core is notified of the results of the payer Payment Service's storage of the Receipt data for the current and previous transactions.

A.4.4 Payee Payment Service primitives

A.4.4.1 PAYEEPS-MakePayment.request

A.4.4.1.1 Function

This primitive requests that a payment be made. The method of generating Receipt data from PaymentMeans data is outside the scope of this standard. For example, it may require communication with a central payment service or an application-specific procedure entirely within the payee application.

A.4.4.1.2 Semantics of the service primitive

The parameters of the PAYEEPSPS-MakePayment.request primitive are as follows:

```
PAYEEPSPS-MakePayment.request (
        PaymentMeansData,
        Receipt1,
        Receipt2,
        VehicleData
        )
```


Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
PaymentMeansData	PaymentMeans	As defined in ISO 14906	Optional. PaymentMeans data
Receipt1	Receipt	As defined in ISO 14906	Optional. The Receipt for the last transaction
Receipt2	Receipt	As defined in ISO 14906	Optional. The Receipt for the penultimate transaction
VehicleData	AttributeList	As defined in ISO 14906	List of attributes containing vehicle related parameters necessary for calculation of the fee

A.4.4.1.3 When generated

This primitive is generated by a payee Application Core when it wishes to make a payment.

A.4.4.1.4 Effect of receipt

The payee Payment Service shall make the payment and return proof of payment in PAYEEPS-MakePayment.confirm.

A.4.4.2 PAYEEPS-MakePayment.confirm

A.4.4.2.1 Function

This primitive reports the results of a payment request.

A.4.4.2.2 Semantics of the service primitive

The parameters of the PAYEEPS-CreateInvoice.confirm primitive are as follows:

```
PAYEEPS-MakePayment.confirm(
        Fee,
        ResultCode,
        Receipt1
        )
```


Copyright © 2011 IEEE. All rights reserved.

Name	Type	Valid range	Description
Fee	PaymentFee	As defined in ISO 14906	Fee to be paid
ResultCode	Enumeration	SUCCESS, INVALID_PARAMETER, INVALID PAYMENT DATA OTHER_ERROR	The success or reason for failure of making the payment
Receipt1	Receipt	As defined in ISO 14906	If ResultCode = SUCCESS, Receipt1 shall contain the Receipt data for the current transaction

A.4.4.2.3 When generated

This primitive is generated by a payee Payment Service as a result of PAYEEPS-MakePayment.request.

A.4.4.2.4 Effect of receipt

The payee Application Core is notified of the results of the payment.

A.5 EFC EPS transactions

The transaction model shall comply with ISO 14906, Clause 6. This transaction model comprises two phases, the initialization phase and the transaction phase.

There are no additional requirements on the transaction phase in this standard. An EFC transaction may be performed in any way, provided that all other requirements in this standard are met.

An EFC transaction proceeds in the following generic sequence of steps:

Initialization phase:

a) Upon receipt of the WME-Notification.indication primitive (as defined in IEEE Std 1609.3-2010) from the WAVE stack indicating the assignment of service channel (SCH) access, the payer WIAL performs necessary application-specific initialization tasks; for example, checking the Provider Service Context (as defined in IEEE Std 1609.3-2010), which contains the BST as noted in A.3.1.

b) The payer WIAL then submits the NotifyActionOBU.indication primitive (as defined in ISO 14906) to the payer Application Core, signifying that communications with an RSU have begun.

c) The payer WIAL then gets the Provider MAC Address of the payee WIAL entity handling peer-to-peer communications by submitting the WME-Get.request primitive (as defined in IEEE Std 1609.3-2010) to the WAVE stack.

d) The payer WIAL prepares the application-specific Vehicle Service Table message (VST, as defined in ISO 14906). This contains an ApplicationList that contains as parameter, the EFC-ContextMark concatenated with a Diversifier corresponding to that EFC-ContextMark, and a Payer Random Value). The payer WIAL obtains the Diversifier by selecting a (KeyRef, KeyInstance) pair and recovering the corresponding DiversifierData using the PS-RetrieveKeyMaterial primitive defined in A.4.2.10, A.4.2.11. The method used to select the (KeyRef, KeyInstance) pair is at the discretion of the implementation. The resulting ApplicationContextMark shall be encoded as follows:

Copyright © 2011 IEEE. All rights reserved.

```
ApplicationContextMark ::= OCTET STRING (CONTAINING Dot11VSTContent)

Dot11VSTContent ::= SEQUENCE {
     eFCContextMark        EFC-ContextMark,
     keyDiversifer         KeyDiversifer,
     rndPayer OCTET STRING -- 16 Bytes
                           }

KeyDiversifier::=     SEQUENCE {
     diversifierData       OCTET STRING (SIZE(16)), -- nonce to diversify the
key
     keyRef                INTEGER (0..255),
     keyInstance           INTEGER (0..255)
                           }
```

The payer WIAL sends the VST to payee by submitting the WSM-WaveShortMessage.request primitive (as defined in IEEE Std 1609.3-2010) using the Provider MAC address. The payer may store multiple Diversifiers corresponding to a given EFC-ContextMark and may vary the Diversifier it uses in the VST.

e) Upon acceptance of WSM-WaveShortMessage.indication (as defined in IEEE Std 1609.3-2010) from the WAVE stack, the payee WIAL shall check that whether the Peer MAC Address parameter does not belong to a peer with which a transaction is currently in progress. If not, this indicates to the payee WIAL that (1) a new payer is applying for a transaction and (2) the message is a VST. The payee WIAL performs necessary application-specific initialization tasks; for example, checking for correctness of the VST message (especially the EFC-ContextMark) or entering the Peer MAC address into a list of currently open transactions.

f) The payee WIAL then submits NotifyActionRSU.indication to the payee Application Core.

Transaction Phase:

g) The payee Application Core then submits a series of request primitives as defined in ISO 15628 and implementing EFC functions as defined in ISO 14906.

h) The payee Application Core may use the ACTION.request primitive to request that specific actions (i.e., EFC functions) be performed by the payer:

   1) The payee Application Core may use the GET_SECURE function to securely retrieve data from the payer including an authenticator over the data, using specified keys. The GET_SECURE request shall include a Payee Random Value and the response shall include an authentication code calculated over the Payee Random Value. The GET_SECURE request shall include a KeyRefEncryption value equal to 105. The Payer shall encrypt the GET_SECURE response using the PS-Encrypt SAP defined in A.4.2.2, A.4.2.3, where KeyRef, KeyInstance are the final two fields in the Diversifier included in the VST. The Payee shall decrypt the GET_SECURE response using the PS-Decrypt SAP defined in A.4.2.4, A.4.2.5, where DiversifierData is the DiversifierData field from the Diversifier included in the VST The authentication code shall be calculated using the PS-CalculateMessageAuthenticationCode SAP defined in A.4.2.6, A.4.2.7, with the DeviceIdentityString defined in A.6.2 used as the Diversifier.

   2) The payee Application Core may use the SET_SECURE function to securely transmit data (for storage on the payer), and retrieve an authenticator over the data, using specified keys and authenticating the SET_SECURE function with data including random data provided by the payer. The SET_SECURE request message shall include an authentication code calculated over the Payer Random Value included in the VST. The Payee shall encrypt the SET_SECURE response using the PS-Encrypt SAP defined in A.4.2.2, A.4.2.3, where DiversifierData is the DiversifierData field from the Diversifier included in the VST. The Payer shall decrypt the SET_SECURE response using the PS-Decrypt SAP defined in

Copyright © 2011 IEEE. All rights reserved.

A.4.2.4, A.4.2.5, where KeyRef, KeyInstance are the final two fields in the Diversifier included in the VST. The authentication code shall be calculated using the PS-CalculateMessageAuthenticationCode SAP defined in A.4.2.6, A.4.2.7, with the DeviceIdentityString defined in A.6.2 used as the Diversifier.

3) The payee Application Core may change or add to the stored (Diversifier, Key) pairs on the payer. In this case it shall select KeyRef and KeyInstance values and use the PS-GenerateKeyMaterial SAP defined in A.4.2.12, A.4.2.13 to generate DiversifierData and the associated key. The new DiversifierData and the new key material shall be set on the Payer using two Attribute fields in a single SET_SECURE message.

i) The payee WIAL encodes these requests into an octet sequence as defined in ISO 15628.

j) The payee WIAL shall send the sequence to the payer as a message by submitting WSM-WaveShortMessage.request to the WAVE stack.

k) Upon acceptance of WSM-WaveShortMessage.indication, the payer WIAL shall then decode the sequence and submit them in a series of indication primitives (as defined in ISO 15628) to the payer Application Core.

l) The payer Application Core uses the payer Payment Service to perform payment operations in support of the payee's requests as defined in A.3.1.

m) The payer WIAL then accepts the requested data from the payer Application Core via a series of response primitives (as defined in ISO 15628).

n) The payer Application Core then submits a series of response primitives (as defined in ISO 15628),

o) The payer WIAL shall send the sequence to the payee as a message by submitting WSM-WaveShortMessage.request to the WAVE stack.

p) Upon acceptance of WSM-WaveShortMessage.indication, the payee WIAL shall then decode the sequence and deliver them in a series of confirm primitives (as defined in ISO 15628) to the payee Application Core.

q) Having received its requested information, the payee Application Core performs application-specific processing, using its Payment Service to perform the necessary payment operations as defined in A.4.

A.6 Keying material

A.6.1 Approved algorithms

A.6.1.1 Encryption

Encryption shall be performed using the AES algorithm in cipher-block chaining (CBC) mode as specified in NIST Special Publication 800-38A 2001, with the provided InitialValue used as the initialization vector (IV). The data shall be padded as necessary using the method described in NIST Special Publication 800-38A 2001, Annex A.

Decryption shall be performed using the AES algorithm in CBC mode as specified in NIST Special Publication 800-38A 2001, with the provided InitialValue used as the IV. The data shall be unpadded using the method described in NIST Special Publication 800-38A 2001, Annex A. In this method, the Payment Service decrypts the data and searches the final block for the last binary 1; it then removes that 1 and all succeeding 0s before returning the decrypted message. If this removal results in a decrypted plaintext that is not an integer number of octets, this shall be considered a decryption error and the calling Application Core shall be notified as such using the confirm primitive.

Copyright © 2011 IEEE. All rights reserved.

No other algorithm is approved for use in this version of this standard.

A.6.1.2 Authentication

The only algorithm approved for authentication in this version of this standard shall be AES-cipher-based message authentication code (CMAC), as specified in NIST Special Publication 800-38B 2005.

A.6.1.3 Key derivation

The only algorithm approved for key derivation in this version of this standard is AES-256 single block encryption. The input data D shall be 16 bytes long. The encryption key K shall be 256 bits (32 bytes) long. The resulting key shall be derived by encrypting D with K and using the output ciphertext as the key.

For encryption, the data D shall be a Diversifier. For authentication, the data D shall be a Device Identity String as described in A.6.2.

A.6.2 Payer

For each supported EFC-ContextMark, a payer Payment Service shall maintain at least the following keys and keying-related material. All of the keys listed next shall be keys for one of the approved algorithms listed above. These keys are not required to be static for the lifetime of the payer device.

— One or more instances of KeyDiversifier to be sent in the VST, associated with the Payment Service Encryption Keys, stored in AttributeId = 99 and distinguished by instance number.

— The encryption keys associated with each Diversifier, stored in AttributeId = 105 and distinguished by instance number.

— A Device Identity String in the form of the EquipmentOBUID (4 Byte) ||Contract Provider (3 Byte) ||ManufacturerID (2 Bytes) ||EquipmentOBUID (4 Byte) ||Contract Provider (3 Byte). This is used as a Diversifier for the authentication keys.

— Authentication Key(s) (with keyRefMAC = 111, …).

The payer Payment Service may maintain multiple keys for authentication if different attributes will be consumed by different entities within the payment processing system.

A payer Application Core shall store the Diversifiers and the Device Identity appropriate to the EFC-ContextMarks that it recognizes, and use them to reference keys within the Payment service.

For each EFC-ContextMark, Encryption keys shall be derived from one single Master key using the key derivation method of A.6.1.3, with the Encryption Master Key used as the key K and the Diversifier used as the input data D.

For each EFC-ContextMark, Authentication keys shall be derived from one single Master key using the key derivation method of A.6.1.3, with the Authentication Master Key used as the key K and the Device Identity String used as the input data D.

A.6.3 Payee

A payer Payment Service shall maintain at least the following keys and keying-related material. All of the keys listed next shall be keys for one of the approved algorithms listed above.

Copyright © 2011 IEEE. All rights reserved.

— A means to map from an EFC-ContextMark to a payer-specific Encryption Master Key.

— A means to map from an EFC-ContextMark to a set of one or more payer-specific Authentication Master Keys.

— A Encryption Master Key.

— Authentication Master Key(s).

For each EFC-ContextMark, Encryption keys shall be derived from one single Master key using the key derivation method of A.6.1.3, with the Encryption Master Key used as the key K and the Diversifier used as the input data D.

For each EFC-ContextMark, Authentication keys shall be derived from one single Master key using the key derivation method of A.6.1.3, with the Authentication Master Key used as the key K and the Device Identity String used as the input data D.

Copyright © 2011 IEEE. All rights reserved.

Annex B

(informative)

Examples of Electronic Fee Collection (EFC) systems per ISO 14906 and ISO 15628

B.1 Introduction

This annex contains examples relevant to Electronic Fee Collection (EFC) systems in which the relationship between an application and its WAVE Interface Application Layer (WIAL) is defined in ISO 14906 and ISO 15628. Requirements for this type of system are defined in Annex A. Hereafter in this annex, all references to EFC refer to instances in this type of EFC system.

B.2 Example EFC transaction

An example EFC transaction is shown in Figure B.1, Figure B.2, and Figure B.3. The use of ReceiptData2 is optional.

The primitives, parameters, and attributes are discussed in Annex A.

Copyright © 2011 IEEE. All rights reserved.

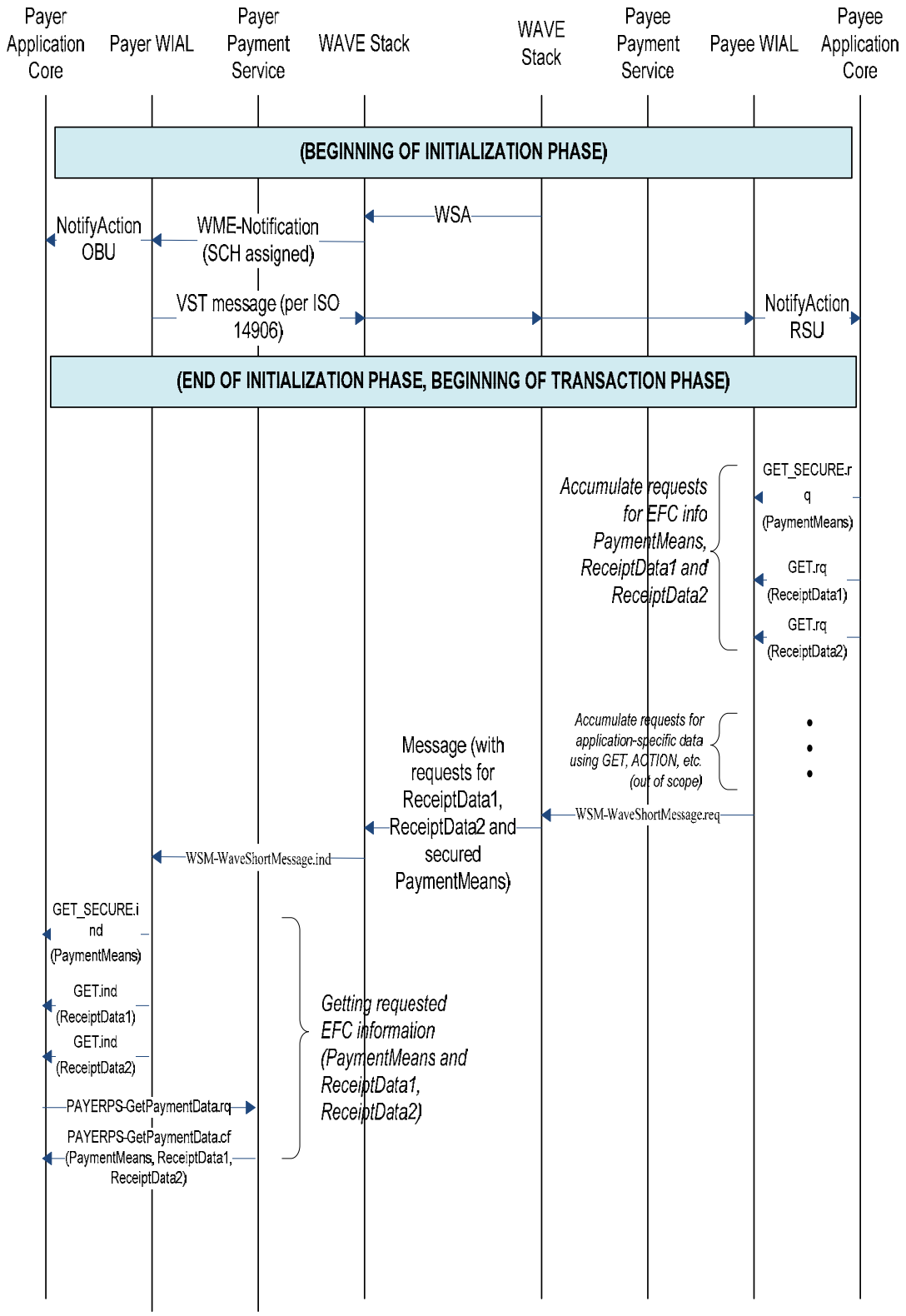

Figure B.1—Example EFC Transaction (part 1)

Copyright © 2011 IEEE. All rights reserved.

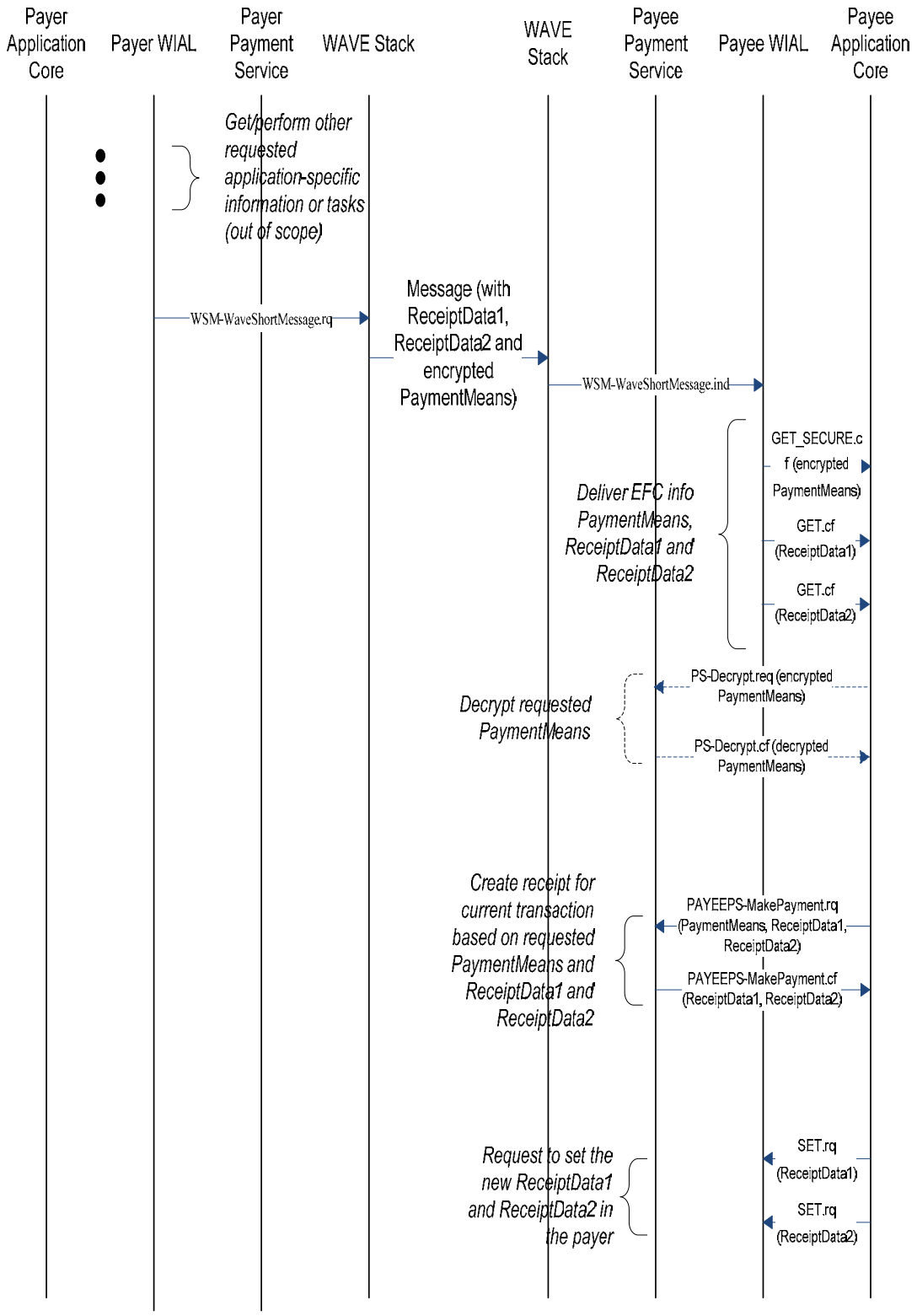

Figure B.2—Example EFC Transaction (part 2)

Copyright © 2011 IEEE. All rights reserved.

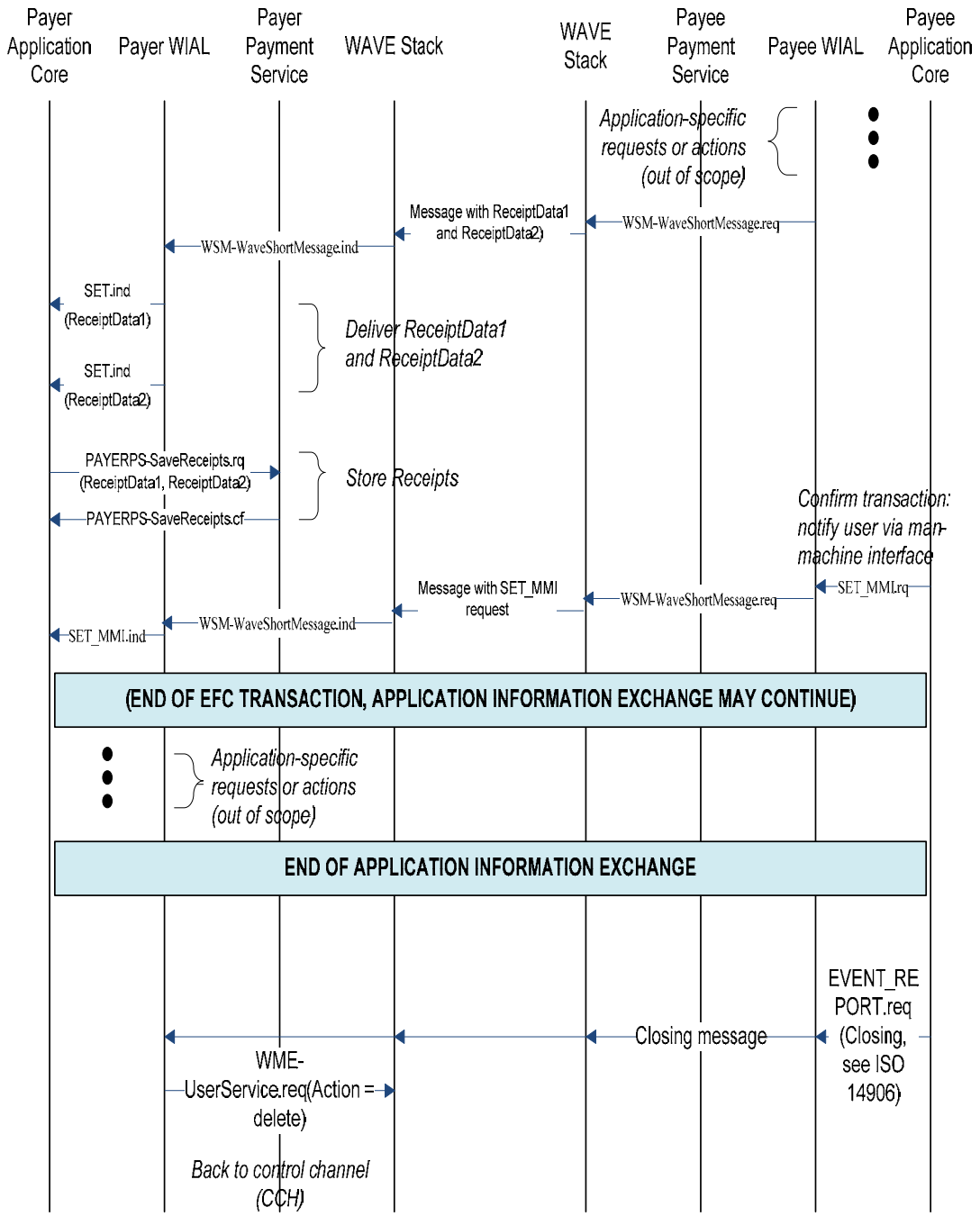

Figure B.3—Example EFC Transaction (part 3)

Copyright © 2011 IEEE. All rights reserved.

B.3 Example of an EFC system that uses small zones and un-augmented location determination, with high-speed payers

B.3.1 Introduction

Subclauses contain an example of an EFC system that is interoperable, supports high-speed payers, uses small zones and has un-augmented location determination. *Small zone* defines that the payee antenna coverage is 30 m or less. *Un-augmented* defines that the location determination of the payer by the payee is obtained solely by the presence of the payer in the coverage of a payee antenna. The combination of high speed and small zone means that the system may not tolerate latencies introduced by channel switching as defined by IEEE Std 1609.4-2010.

B.3.2 Payer and payee approval types

The example system uses the following payer and payee approval types:

— *Payer response pre-approval* as described in 6.2.2.

— *Payee action pre-approval* as described in 6.2.3.

— *Payer action pre-approval* as described in 6.2.4.

B.3.3 Payee WIAL

This subclause describes the actions of the Payee WIAL, in addition to the requirements in A.3.4, for the example system.

Upon acceptance of RegisterApplicationRSU.request (as defined in ISO 14906) from the Payee Application Core, the payee WIAL requests WAVE provider services by submitting the WME-ProviderService.request (as defined in IEEE Std 1609.3-2010) with parameters set as follows:

— ProviderServiceContext set as described in A.2

— Action = Add

— IP Service = false

The other parameters of WME-ProviderService.request are set as required by the application.

Upon acceptance of DeregisterApplicationRSU.request (as defined in ISO 14906) from the Payee Application Core, the payee WIAL deletes the WAVE provider service by submitting the WME-ProviderService.request (as defined in IEEE Std 1609.3-2010) with parameter Action set to Delete:

The WAVE stack will then stop broadcasting WSAs, prohibiting subsequent EFC and application-specific transactions.

B.3.4 Payer WIAL

This subclause describes the actions of the Payer WIAL, in addition to the requirements in A.3.5, for the example system.

Copyright © 2011 IEEE. All rights reserved.

Upon acceptance of RegisterApplicationOBU.request (as defined in ISO 14906) from the payer Application Core, the payer WIAL requests WAVE user services by submitting the WME-UserService.request (as defined in IEEE Std 1609.3-2010) with parameters set as follows:

— ImmediateAccess = true

— ExtendedAccess = true

— Action = Add

The other parameters of WME-UserService.request may be set as required by the application.

The WAVE stack will then listen for WSA messages.

Upon acceptance of DeregisterApplicationOBU.request (as defined in ISO 14906) from the Payer Application Core, the payer WIAL deletes the WAVE user service by submitting the WME-UserService.request (as defined in IEEE Std 1609.3-2010) with parameter Action set to Delete.

The WAVE stack will then stop listening for WSA messages

B.3.5 EPS EFC transaction guidelines

The following are guidelines for the example system:

— The payee Application Core retrieves those vehicle related attributes from the payer Application Core which are needed for the calculation of the fee according to the tariff table and which are not already available at the payee.

— The payee's Application Core retrieves the attribute VehicleLicencePlateNumber[9] for compliance checking purposes if necessary.

— The payee Application Core uses the attribute EquipmentStatus, as defined in ISO 14906, as a transaction counter which is read-out, incremented, and written back by the payee Application Core at each transaction and stored by the payer Application Core. The EquipmentStatus is read out and authenticated together with the PaymentMeans.

— The payee Application Core uses the attribute EquipmentOBUuId, as defined in ISO 14906, in order to identify the equipment for further compliance checking purposes. The EquipmentOBUID may also be used for authentication of the equipment.

B.3.6 Payer application data

For the example system, the Payer Application Core implements at least the following attributes:

Name	AttributeId	Container Type	Description
VehicleLicencePlateNumber	16	47	As defined in ISO 14906
VehicleClass	17	49	As defined in ISO 14906
VehicleAxles	19	51	As defined in ISO 14906
VehicleWeightLimits	20	52	As defined in ISO 14906
DriverCharacteristic	27	59	As defined in ISO 14906

The payer Application Core also supports as many attributes of the data group VEHICLE, as defined in ISO 14906, as possible to maximize the probability that any payee Application Core can calculate the fee (i.e., the amount to be paid) according to its tariff scheme.

[9] The spelling of "licence" is as given in ISO 14906.

Copyright © 2011 IEEE. All rights reserved.

In order to correctly support the above guidelines, the payer Application Core implements the following attributes:

Name	AttributeId	Container Type	Description
EquipmentOBUId	24	56	As defined in ISO 14906, 4 Bytes length
EquipmentStatus	26	58	As defined in ISO 14906

The payer Application Core could also support the current GPS position of the OBU (if available) and other "private" attributes as defined in ISO 14906.

B.3.7 Example Protocol Implementation Conformance Statement (PICS) proforma

The table below contains an example protocol implementation conformance statement (PICS) proforma for an application using the Payment Service as described in this annex. The columns and symbols used in the table are described in Annex C.

Item	Feature	Value	Reference	Status	Conformance
F1	**EFC systems that use small zones and un-augmented location determination, with high-speed payees**		B.3	E2	
F1.1	Payer response approval type		A.1	M	
F1.2	Payee action approval type		A.1	M	
F1.3	Payer action approval type		A.1	M	
F1.4	**Payer WIAL**		—	—	
F1.4.1	Immediate SCH access		B.3.4	M	
F1.4.2	Extended SCH access		B.3.4	M	
F1.5	**EFC transaction guidelines**		—	—	
F1.5.1	Payee application core retrieves needed vehicle related attributes		B.3.5	M	
F1.5.2	Payee application core retrieves VehicleLicencePlateNumber		B.3.5	M	
F1.5.3	Payee application core usage of EquipmentStatus		B.3.5	M	
F1.5.4	Payee application core usage of EquipmentOBUuId		B.3.5	M	
F1.6	**Payer application core supported attributes**		—	—	
F1.6.1	VehicleClass attribute		B.3.6	M	

Copyright © 2011 IEEE. All rights reserved.

Item	Feature	Value	Reference	Status	Conformance
F1.6.2	VehicleAxles attribute		B.3.6	M	
F1.6.3	VehicleWeightLimits attribute		B.3.6	M	
F1.6.4	DriverCharacteristic attribute		B.3.6	M	
F1.6.5	EquipmentOBUId		B.3.6	M	
F1.6.6	EquipmentStatus		B.3.6	M	

Copyright © 2011 IEEE. All rights reserved.

Annex C

(normative)

Protocol Implementation Conformance Statement (PICS) proforma[10]

NOTES:

— Entries in the **Item** column may be hierarchical. Thus an entry of the form E<a>.<b> indicates the item is part of the group identified by the item E<a> where all members of the group are subject to the conditions of applicability of E<a>, i.e., features lower in the numbering hierarchy (E<a>.<b>) are only applicable if the next higher level feature (E<a>) is identified in the C**onformance** column as being present.

— Parentheses in the **Value** column indicate the user should enter information as specified in the accompanying footnote.

— An entry of the form <pred>:<S> in the **Status** column indicates that the status <S> applies if the item identified by <pred> is identified in the C**onformance** column as being present.

— Valid status values in the **Status** column are M, O, O<n> and C<n>.

  — A status of M indicates a mandatory feature.

  — A status of O indicates an optional feature.

  — A status of O<n> indicates a mutual conditionality such that the feature is optional but that support of at least one of the items that have status O<n> is mandatory.

— A status of C<n> indicates a mutual conditionality such that support of one and one only of the items that have status C<n> is mandatory.

Item	Feature	Value	Reference	Status	Conformance
E1	**General**		—	—	
E1.1	**Type of systems using EPS**		—	—	
E1.1.1	**EPS Information Exchange Modes**		—	—	
E1.1.1.1	Direct EPS Communication Mode		6.1.2	O1	
E1.1.1.2	Indirect EPS Communication Mode		6.1.3	O1	
E1.2	**System operation types**		—	—	
E1.2.1	EFC per ISO 14906 and ISO 15628 supported		—	O	
E2	**Electronic Fee Collection (EFC) systems per ISO 14906 and ISO 15628**		Annex A	E1.2.1:M	
E2.1	EPS information exchange mode		A.1	E1.1.1.2:M	
E2.2	WIAL-to-Application Core interface as specified in ISO 14906 and ISO 15628		A.2	M	
E2.3	WIAL-to-WAVE interface per IEEE Std 1609.3-2010		A.2	M	

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

Copyright © 2011 IEEE. All rights reserved.

Item	Feature	Value	Reference	Status	Conformance
E2.4	Message Authentication Code per AES-128 and FIPS 197		A.2	M	
E2.5	PSC contents		A.2	M	
E2.6	MAC replacement of LID		A.2	M	
E2.7	WIAL functions supported		A.3	M	
E2.8	EFC functions supported		A.3	M	
E2.9	GET_SECURE format		A.3.2	M	
E2.10	SET_SECURE format		A.3.3	M	
E2.11	**Security**		—	—	
E2.11.1	Encryption method		A.6.1.1	M	
E2.11.2	Decryption method		A.6.1.1	M	
E2.11.3	Authentication method		A.6.1.2	M	
E2.11.4	Key derivation		A.6.1.3	M	
E2.11.4.1	Key derivation for encryption		A.6.1.3	E2.11.1:M	
E2.11.4.2	Key derivation for authentication		A.6.1.3	E2.11.3:M	
E2.12	**Payer Payment Service, for each supported EFC-ContextMark**		—	—	
E2.12.1	Diversifiers associated with Payment Service Encryption Key(s)	99	A.6.2	M	
E2.12.2	Device Identity String		A.6.2	M	
E2.12.3	Encryption Key(s)	105	A.6.2	M	
E2.12.4	Authentication Key(s)	111	A.6.2	M	
E2.12.5	Encryption keys derived from one single Master key		A.6.1.3, A.6.2	M	
E2.12.6	Authentication keys derived from one single Master key		A.6.1.3, A.6.2	M	
E2.13	**Payee Payment Service**		—	—	
E2.13.1	Mapping from EFC-ContextMark to a payer-specific Encryption Master Key		A.6.3	M	
E2.13.2	Mapping from EFC-ContextMark to a set of one or more payer-specific Authentication Master Key		A.6.3	M	
E2.13.3	Encryption Master Key(s)		A.6.3	M	
E2.13.4	Authentication Master Key(s)		A.6.3	M	

Copyright © 2011 IEEE. All rights reserved.