



BSI Standards Publication

Electricity metering data exchange — The DLMS/COSEM suite

Part 5-3: DLMS/COSEM application layer

National foreword

This British Standard is the UK implementation of EN 62056-5-3:2014. It is identical to IEC 62056-5-3:2013. Together with BS EN 62056-1-0, BS EN 62056-7-6:2013 and BS EN 62056-9-7:2013 it supersedes BS EN 62056-53:2007, which will be withdrawn upon publication of BS EN 62056-1-0.

The UK participation in its preparation was entrusted to Technical Committee PEL/13, Electricity Meters.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2014.
Published by BSI Standards Limited 2014

ISBN 978 0 580 74606 2
ICS 17.220; 35.110; 91.140.50

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 June 2014.

Amendments/corrigenda issued since publication

Date	Text affected
------	---------------

**Electricity metering data exchange -
The DLMS/COSEM suite -
Part 5-3: DLMS/COSEM application layer
(IEC 62056-5-3:2013)**

Échange des données de comptage de
l'électricité -
La suite DLMS/COSEM -
Partie 5-3: Couche application
DLMS/COSEM
(CEI 62056-5-3:2013)

Datenkommunikation der elektrischen
Energienmessung - DLMS/COSEM -
Teil 5-3: DLMS/COSEM-
Anwendungsschicht
(IEC 62056-5-3:2013)

This European Standard was approved by CENELEC on 2013-07-10. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the CEN-CENELEC Management Centre or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the CEN-CENELEC Management Centre has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, Former Yugoslav Republic of Macedonia, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland, Turkey and the United Kingdom.

CENELEC

European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

CEN-CENELEC Management Centre: Avenue Marnix 17, B - 1000 Brussels

Foreword

The text of document 13/1523/FDIS, future edition 1 of IEC 62056-5-3, prepared by IEC/TC 13 "Electrical energy measurement, tariff- and load control" was submitted to the IEC-CENELEC parallel vote and approved by CENELEC as EN 62056-5-3:2014.

The following dates are fixed:

- latest date by which the document has to be (dop) 2014-09-07
implemented at national level by
publication of an identical national
standard or by endorsement
- latest date by which the national (dow) 2016-07-10
standards conflicting with the
document have to be withdrawn

This document supersedes EN 62056-53:2007 (PART).

EN 62056-5-3:2014 includes the following significant technical changes with respect to EN 62056-53:2007:

The significant technical changes with respect to EN 62056-53 are listed in Annex F.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CENELEC [and/or CEN] shall not be held responsible for identifying any or all such patent rights.

Endorsement notice

The text of the International Standard IEC 62056-5-3:2013 was approved by CENELEC as a European Standard without any modification.

In the official version, for Bibliography, the following notes have to be added for the standards indicated:

IEC 61334-4-32:1996	NOTE	Harmonized as EN 61334-4-32:1996 (not modified).
IEC 61334-4-511:2000	NOTE	Harmonized as EN 61334-4-511:2000 (not modified).
IEC 61334-4-512:2001	NOTE	Harmonized as EN 61334-4-512:2002 (not modified).
IEC 61334-5-1:2001	NOTE	Harmonized as EN 61334-5-1:2001 (not modified).
IEC 62056-1-0 ¹⁾	NOTE	Harmonized as EN 62056-1-0 ¹⁾ .
ISO/IEC 7498-1:1994	NOTE	Harmonized as EN ISO/IEC 7498-1:1995 ²⁾ (not modified).

¹⁾ At draft stage.

²⁾ Withdrawn publication.

Annex ZA (normative)

Normative references to international publications with their corresponding European publications

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

NOTE When an international publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
IEC 61334-4-41	1996	Distribution automation using distribution line carrier systems - Part 4: Data communication protocols - Section 41: Application protocols - Distribution line message specification	EN 61334-4-41	1996
IEC 61334-6	2000	Distribution automation using distribution line carrier systems - Part 6: A-XDR encoding rule	EN 61334-6	2000
IEC/TR 62051	1999	Electricity metering - Glossary of terms	-	-
IEC/TR 62051-1 + corr. June	2004 2005	Electricity metering - Data exchange for meter reading, tariff and load control - Glossary of terms - Part 1: Terms related to data exchange with metering equipment using DLMS/COSEM	-	-
IEC 62056-6-1	2013	Electricity metering data exchange - The DLMS/COSEM suite - Part 6-1: COSEM Object Identification System (OBIS)	EN 62056-6-1	2013
IEC 62056-6-2	2013	Electricity metering data exchange - The DLMS/COSEM suite - Part 6-2: COSEM interface classes	EN 62056-6-2	2013
IEC 62056-8-3	2013	Electricity metering data exchange - The DLMS/COSEM suite - Part 8-3: Communication profile for PLC S-FSK neighbourhood networks	EN 62056-8-3	2013
ISO/IEC 15953	1999	Information technology - Open systems interconnection - Service definition for the Application service object association control service element	-	-
ISO/IEC 15954	1999	Information technology - Open systems interconnection - Connection-mode protocol for the application service object association control service element	-	-
ISO/IEC 8824-1	2008	Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation	-	-
ISO/IEC 8825-1	2008	Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)	-	-

<u>Publication</u>	<u>Year</u>	<u>Title</u>	<u>EN/HD</u>	<u>Year</u>
FIPS PUB 180-1	1995 ³⁾	Secure hash standard	-	-
FIPS PUB 197	2001	Advanced Encryption Standard (AES)	-	-
NIST SP 800-38D	2007	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC	-	-
NIST SP 800-57	2007 ⁴⁾	Recommendation for key management - Part 1: General	-	-
RFC 1321	1992	Internet Engineering Task Force (IETF). The MD5 Message-Digest Algorithm. Edited by R. Rivest (MIT Laboratory for Computer Science and RSA Data Security, Inc.)	-	-
RFC 3394	2002	Internet Engineering Task Force (IETF). Advanced Encryption Standard (AES) Key Wrap Algorithm. Edited by J. Schaad (Soaring Hawk Consulting) and R. Housley (RSA Laboratories)	-	-
RFC 4106	2005	The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)	-	-

³⁾ Superseded by FIPS PUB 180-2:2002, which is also superseded by FIPS PUB 180-4:2012.

⁴⁾ Superseded by NIST SP 800-57:2012.

CONTENTS

1	Scope	9
2	Normative references	9
3	Terms, definitions and abbreviations	11
3.1	Terms and definitions	11
3.2	Abbreviations	11
4	Overview	13
4.1	COSEM application layer structure	13
4.2	COSEM application layer services	14
4.2.1	ASO services	14
4.2.2	Services provided for application association establishment and release	14
4.2.3	Services provided for data transfer	15
4.2.4	Layer management services	19
4.2.5	Summary of COSEM application layer services	19
4.3	COSEM application layer protocols	20
5	Information security in DLMS/COSEM	20
5.1	Definitions	20
5.2	General	20
5.3	Data access security	21
5.3.1	Overview	21
5.3.2	Lowest level security (no security)	21
5.3.3	Low Level Security (LLS)	21
5.3.4	High Level Security (HLS)	22
5.4	Data transport security	23
5.4.1	Applying, removing or checking the protection: ciphering and deciphering	23
5.4.2	Security context	25
5.4.3	Security policy	25
5.4.4	Security suite	25
5.4.5	Security material	26
5.4.6	Ciphered xDLMS APDUs	26
5.4.7	Cryptographic keys	27
5.4.8	The Galois/Counter Mode of Operation (GCM)	30
6	COSEM application layer service specification	40
6.1	Service primitives and parameters	40
6.2	The COSEM-OPEN service	42
6.3	The COSEM-RELEASE service	47
6.4	COSEM-ABORT service	49
6.5	Security parameters	50
6.6	The GET service	51
6.7	The SET service	53
6.8	The ACTION service	56
6.9	The EventNotification service	60
6.10	The TriggerEventNotificationSending service	61
6.11	Variable access specification	62
6.12	The Read service	63

6.13	The Write service	67
6.14	The UnconfirmedWrite service.....	70
6.15	The InformationReport service.....	71
6.16	Client side layer management services: the SetMapperTable.request.....	72
6.17	Summary of services and LN/SN data transfer service mapping	72
7	COSEM application layer protocol specification	73
7.1	The control function.....	73
7.1.1	State definitions of the client side control function	73
7.1.2	State definitions of the server side control function	75
7.2	The ACSE services and APDUs	76
7.2.1	ACSE functional units, services and service parameters.....	76
7.2.2	Registered COSEM names	79
7.2.3	APDU encoding rules	81
7.2.4	Protocol for application association establishment.....	81
7.2.5	Protocol for application association release	85
7.3	Protocol for the data transfer services	89
7.3.1	Negotiation of services and options – the conformance block	89
7.3.2	Confirmed and unconfirmed service invocations	90
7.3.3	Protocol for the GET service.....	91
7.3.4	Protocol for the SET service	94
7.3.5	Protocol for the ACTION service.....	97
7.3.6	Protocol for the EventNotification service	99
7.3.7	Protocol for the Read service	100
7.3.8	Protocol for the Write service.....	104
7.3.9	Protocol for the UnconfirmedWrite service	108
7.3.10	Protocol for the InformationReport service	109
8	Abstract syntax of ACSE and COSEM APDUs	110
Annex A (normative)	Using the COSEM application layer in various communications profiles	124
Annex B (informative)	AARQ and AARE encoding examples.....	126
Annex C (informative)	Encoding examples: AARQ and AARE APDUs using a ciphered application context.....	140
Annex D (informative)	Data transfer service examples	148
Annex E (informative)	Overview of cryptography.....	163
Annex F (informative)	Significant technical changes with respect to IEC 62056-53	169
Bibliography		172
Index		174
Figure 1	– Structure of the COSEM Application layers	13
Figure 2	– Summary of COSEM AL services	19
Figure 3	– LLS and HLS authentication	23
Figure 4	– Data transport security in DLMS/COSEM	24
Figure 5	– Ciphered xDLMS APDUs.....	26
Figure 6	– Cryptographic protection of xDLMS APDUs using GCM.....	33
Figure 7	– Service primitives.....	40
Figure 8	– Time sequence diagrams	41
Figure 9	– Partial state machine for the client side control function	74

Figure 10 – Partial state machine for the server side control function.....	75
Figure 11 – MSC for successful AA establishment preceded by a successful lower layer connection establishment	82
Figure 12 – Graceful AA release using the A-RELEASE service.....	87
Figure 13 – Graceful AA release by disconnecting the supporting layer	88
Figure 14 – Aborting an AA following a PH-ABORT.indication	89
Figure 15 – MSC of the GET service.....	92
Figure 16 – MSC of the GET service with block transfer	92
Figure 17 – MSC of the GET service with block transfer, long GET aborted	94
Figure 18 – MSC of the SET service	95
Figure 19 – MSC of the SET service with block transfer	96
Figure 20 – MSC of the ACTION service.....	98
Figure 21 – MSC of the ACTION service with block transfer.....	99
Figure 22 – MSC of the Read service used for reading an attribute.....	102
Figure 23 – MSC of the Read service used for invoking a method.....	103
Figure 24 – MSC of the Read Service used for reading an attribute, with block transfer	104
Figure 25 – MSC of the Write service used for writing an attribute	107
Figure 26 – MSC of the Write service used for invoking a method.....	107
Figure 27 – MSC of the Write Service used for writing an attribute, with block transfer	108
Figure 28 – MSC of the Unconfirmed Write service used for writing an attribute.....	109
Figure E.1 – Hash function.....	164
Figure E.2 – Encryption and decryption.....	165
Figure E.3 – Message Authentication Codes (MACs)	166
Table 1 – Clarification of the meaning of PDU Size for DLMS/COSEM	16
Table 2 – Security suites	26
Table 3 – Security control field.....	27
Table 4 – Cryptographic keys and their management.....	30
Table 5 – Plaintext and additional authenticated data	34
Table 6 – Example for ciphered APDUs	37
Table 7 – HLS example with GMAC	39
Table 8 – Codes for AL service parameters	42
Table 9 – Service parameters of the COSEM-OPEN service primitives	43
Table 10 – Service parameters of the COSEM-RELEASE service primitives	47
Table 11 – Service parameters of the COSEM-ABORT service primitives	50
Table 12 – Security parameters	50
Table 13 – Service parameters of the GET service	51
Table 14 – GET service request and response types	52
Table 15 – Service parameters of the SET service.....	54
Table 16 – SET service request and response types.....	55
Table 17 – Service parameters of the ACTION service	57
Table 18 – ACTION service request and response types	58
Table 19 – Service parameters of the EventNotification service primitives	60

Table 20 – Service parameters of the TriggerEventNotificationSending.request service primitive	61
Table 21 – Variable Access Specification	63
Table 22 – Service parameters of the Read service	64
Table 23 – Use of the Variable_Access_Specification variants and the Read.response choices	65
Table 24 – Service parameters of the Write service	68
Table 25 – Use of the Variable_Access_Specification variants and the Write.response choices	68
Table 26 – Service parameters of the UnconfirmedWrite service	70
Table 27 – Use of the Variable_Access_Specification variants	71
Table 28 – Service parameters of the InformationReport service	72
Table 29 – Service parameters of the SetMapperTable.request service primitives	72
Table 30 – Summary of ACSE services	73
Table 31 – Summary of xDLMS services for LN referencing	73
Table 32 – Summary of xDLMS services for SN referencing	73
Table 33 – ACSE functional units, services and service parameters	77
Table 34 – Use of ciphered / unciphered APDUs	80
Table 35 – xDLMS Conformance block	90
Table 36 – GET service types and APDUs	91
Table 37 – SET service types and APDUs	95
Table 38 – ACTION service types and APDUs	98
Table 39 – Mapping between the GET and the Read services	100
Table 40 – Mapping between the ACTION and the Read services	101
Table 41 – Mapping between the SET and the Write services	105
Table 42 – Mapping between the ACTION and the Write service	106
Table 43 – Mapping between the SET and the UnconfirmedWrite services	108
Table 44 – Mapping between the ACTION and the UnconfirmedWrite services	109
Table 45 – Mapping between the EventNotification and InformationReport services	110
Table B.1 – Conformance block	127
Table B.2 – A-XDR encoding of the xDLMS InitiateRequest APDU	128
Table B.3 – A-XDR encoding of the xDLMS InitiateResponse APDU	129
Table B.4 – BER encoding of the AARQ APDU	132
Table B.5 – Complete AARQ APDU	134
Table B.6 – BER encoding of the AARE APDU	135
Table B.7 – The complete AARE APDU	139
Table C.1 – A-XDR encoding of the xDLMS InitiateRequest APDU	140
Table C.2 – Authenticated encryption of the xDLMS InitiateRequest APDU	141
Table C.3 – BER encoding of the AARQ APDU	142
Table C.4 – A-XDR encoding of the xDLMS InitiateResponse APDU	143
Table C.5 – Authenticated encryption of the xDLMS InitiateResponse APDU	144
Table C.6 – BER encoding of the AARE APDU	145
Table C.7 – BER encoding of the RLRQ APDU	146
Table C.8 – BER encoding of the RLRE APDU	

Table D.1 – Objects used in the examples	148
Table D.2 – Example: Reading the value of a single attribute without block transfer	149
Table D.3 – Example: Reading the value of a list of attributes without block transfer	150
Table D.4 – Example: Reading the value of a single attribute with block transfer	151
Table D.5 – Example: Reading the value of a list of attributes with block transfer	153
Table D.6 – Example: Writing the value of a single attribute without block transfer	155
Table D.7 – Example: Writing the value of a list of attributes without block transfer	156
Table D.8 – Example: Writing the value of a single attribute with block transfer	158
Table D.9 – Example: Writing the value of a list of attributes with block transfer	160

ELECTRICITY METERING DATA EXCHANGE – THE DLMS/COSEM SUITE –

Part 5-3: DLMS/COSEM application layer

1 Scope

This part of IEC 62056 specifies the DLMS/COSEM application layer in terms of structure, services and protocols for COSEM clients and servers, and defines how to use the DLMS/COSEM application layer in various communication profiles.

It defines services for establishing and releasing application associations, and data communication services for accessing the methods and attributes of COSEM interface objects, defined in IEC 62056-6-2², using either logical name (LN) or short name (SN) referencing.

Annex A (normative) defines how to use the COSEM application layer in various communication profiles. It specifies how various communication profiles can be constructed for exchanging data with metering equipment using the COSEM interface model, and what are the necessary elements to specify in each communication profile. The actual, media-specific communication profiles are specified in separate parts of the IEC 62056 series.

Annex B, Annex C and Annex D (informative) include encoding examples for APDUs.

Annex E (informative) provides an overview of cryptography.

2 Normative references

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

IEC 61334-4-41:1996, *Distribution automation using distribution line carrier systems – Part 4: Data communication protocols – Section 41: Application protocols – Distribution line message specification*

IEC 61334-6:2000, *Distribution automation using distribution line carrier systems – Part 6: A-XDR encoding rule*

IEC/TR 62051:1999, *Electricity metering – Glossary of terms*

IEC/TR 62051-1:2004, *Electricity metering – Data exchange for meter reading, tariff and load control – Glossary of terms – Part 1: Terms related to data exchange with metering equipment using DLMS/COSEM*

² To be published simultaneously with this part of IEC 62056.

IEC 62056-6-1:—, *Electricity metering data exchange – The DLMS/COSEM suite – Part 6-1: Object Identification System (OBIS)*³

IEC 62056-6-2:—, *Electricity metering data exchange – The DLMS/COSEM suite – Part 6-2: COSEM interface classes*⁴

IEC 62056-8-3:—, *Electricity metering data exchange – The DLMS/COSEM suite – Part 8-3: Communication profile for PLC S-FSK neighbourhood networks*⁵

ISO/IEC 15953:1999, *Information technology – Open Systems Interconnection – Service definition for the Application Service Object Association Control Service Element*

ISO/IEC 15954:1999, *Information technology – Open Systems Interconnection – Connection-mode protocol for the Application Service Object Association Control Service Element*

ISO/IEC 8824-1:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 8825-1:2008, *Information technology – ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)*

FIPS PUB 180-1:2002, *Secure hash standard*

FIPS PUB 197:2001, *Advanced Encryption Standard (AES)*

NIST SP 800-38D:2007, *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*

NIST SP 800-57:2006, *Recommendation for Key Management – Part 1: General* (Revised)

RFC 1321:1992, Internet Engineering Task Force (IETF). *The MD5 Message-Digest Algorithm*. Edited by R. Rivest (MIT Laboratory for Computer Science and RSA Data Security, Inc.) April 1992. Available from: <http://www.rfc-editor.org/rfc/rfc1321.txt>

RFC 3394:2002, Internet Engineering Task Force (IETF). *Advanced Encryption Standard (AES) Key Wrap Algorithm*. Edited by J. Schaad (Soaring Hawk Consulting) and R. Housley (RSA Laboratories) September 2002. Available from: <http://www.rfc-editor.org/rfc/rfc3394.txt>

RFC 4106:2005, *The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)*

NOTE See also the Bibliography.

³ To be published simultaneously with this part of IEC 62056.

⁴ To be published simultaneously with this part of IEC 62056.

⁵ To be published simultaneously with this part of IEC 62056.

3 Terms, definitions and abbreviations

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in RFC 4106, IEC/TR 62051 and IEC/TR 62051-1 and the following apply.

3.2 Abbreviations

AA	Application Association
AAD	Additional Authenticated Data (used with ciphering of APDUs)
AARE	A-Associate Response – an APDU of the ACSE
AARQ	A-Associate Request – an APDU of the ACSE
ACPM	Association Control Protocol Machine
ACSE	Association Control Service Element
AE	Application Entity
AES	Advanced Encryption Standard
AL	Application Layer
AP	Application Process
APDU	Application Layer Protocol Data Unit
API	Application Programming Interface
ASE	Application Service Element
ASO	Application Service Object
A-XDR	Adapted Extended Data Representation
base_name	The short_name corresponding to the first attribute (“logical_name”) of a COSEM object
BER	Basic Encoding Rules
CF	Control Function
CL	Connectionless
Client	A station, asking for services. In the case of the 3-layer, CO HDLC based profile it is the master station
.cnf	.confirm service primitive
CO	Connection-oriented
COSEM	Companion Specification for Energy Metering
COSEM class_id	COSEM interface class identifier
COSEM object	An instance of a COSEM interface class
DCS	Data Collection System
DLMS	Device Language Message Specification
DLMS UA	DLMS User Association
FCS	Frame Check Sequence
FIPS	Federal Information Processing Standard
GCM	Galois/Counter Mode, an algorithm for authenticated encryption with associated data
GMAC	A specialization of GCM for generating a message authentication code (MAC) on data that is not encrypted
HDLC	High-level Data Link Control

HLS	High Level Security
HMAC	Keyed-Hash Message Authentication Code
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
.ind	.indication service primitive
IP	Internet Protocol
ISO	International Organization for Standardization
IV	Initialization Vector
LDN	Logical Device Name
LLC	Logical Link Control (Sublayer)
LLS	Low Level Security
LSB	Least Significant Bit
MAC	Medium Access Control (sublayer)
MAC	Message Authentication Code (cryptography)
master	Central station – station which takes the initiative and controls the data flow
MSB	Most Significant Bit
MSC	Message Sequence Chart
NIST	National Institute of Standards and Technology
OBIS	OBject Identification System
OSI	Open System Interconnection
PDU	Protocol Data Unit
PHSDU	PH SDU
PLC	Power line carrier
PPP	Point-to-Point Protocol
.req	.request service primitive
.res	.response service primitive
RLRE	A-Release Response – an APDU of the ACSE
RLRQ	A-Release Request – an APDU of the ACSE
SAP	Service Access Point
Server	A station, delivering services. The tariff device (meter) is normally the server, delivering the requested values or executing the requested tasks.
TCP	Transmission Control Protocol
TDEA	Triple Data Encryption Algorithm
UDP	User Datagram Protocol
VAA	Virtual Application Association
xDLMS_ASE	Extended DLMS Application Service Element

4 Overview

4.1 COSEM application layer structure

The structure of the client and server COSEM application layers is shown in Figure 1.

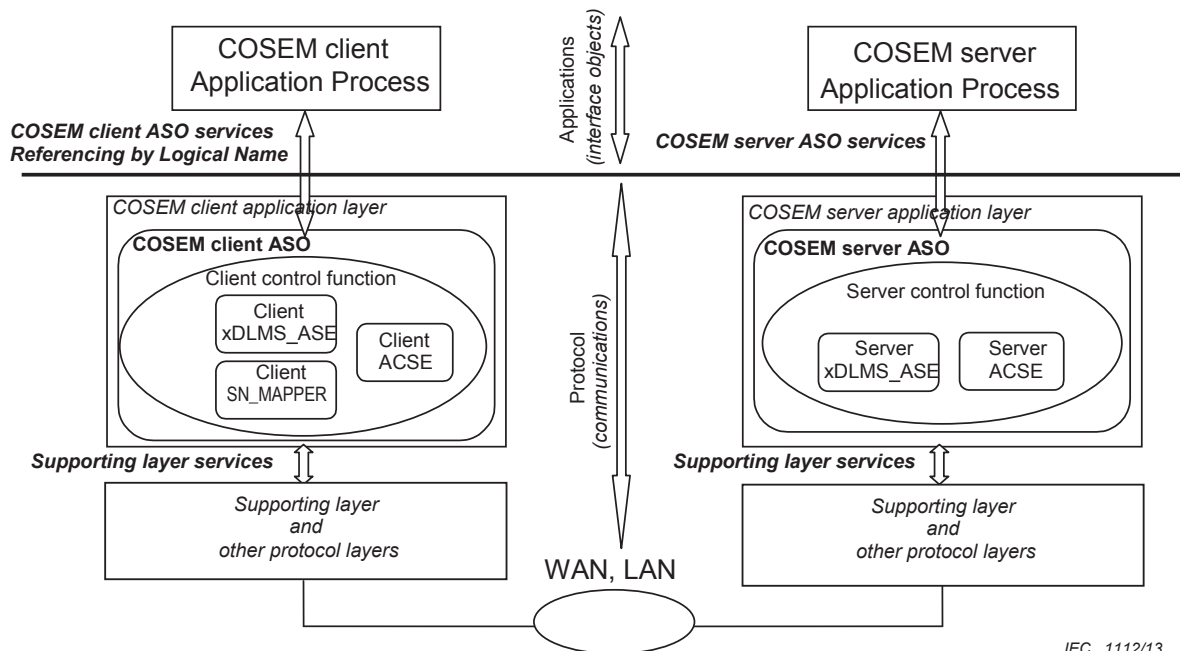


Figure 1 – Structure of the COSEM Application layers

The main component of the COSEM AL is the COSEM Application Service Object. It provides services to its service user, the COSEM Application Process, and uses services provided by the supporting lower layer. It contains three mandatory components both on the client and on the server side:

- the Association Control Service Element, ACSE;
- the extended DLMS Application Service Element, xDLMS_ASE;
- the Control Function, CF.

On the client side, there is a fourth, optional element, called the SN_MAPPER ASE.

The task of the ACSE is to establish, maintain, and release application associations. For the purposes of DLMS/COSEM connection oriented (CO) communication profiles, the CO ACSE, specified in ISO/IEC 15953 and ISO/IEC 15954 is used.

The task of the xDLMS_ASE is to provide data transfer services between COSEM APs. It is based on the DLMS standard, IEC 61334-4-41. It has been extended for DLMS/COSEM; see 4.2.3.1. The main objective of DLMS/COSEM is to provide a business domain oriented interface object model for metering devices and systems while keeping backward compatibility with the DLMS standard. To meet these objectives, DLMS/COSEM includes an evolution of DLMS. Remaining fully compliant to the DLMS standard, DLMS/COSEM provides a more metering specific view of the meter through the COSEM interface objects.

IEC 62056-6-2:—, 4.2 specifies two referencing methods to attributes and methods of COSEM interface objects for use in COSEM servers: LN and SN referencing. When LN referencing is used, the Logical Name of the COSEM objects shall be as specified in IEC 62056-6-1. Therefore, on the server side, two distinct xDLMS service sets are specified: one exclusively

for LN referencing and the other exclusively for SN referencing. It can be considered that there are two different xDLMS_ASEs: one providing services for LN referencing and the other for SN referencing. The server AL may include one, the other or both xDLMS_ASEs.

The CF element specifies how the ASO services invoke the appropriate service primitives of the ACSE, the xDLMS_ASE and the services of the supporting layer.

NOTE Both the client and the server COSEM ASO can contain other, optional application protocol components.

The optional SN_MAPPER ASE is present in the client side AL ASO, when the server uses SN referencing. It provides mapping between services using LN and SN referencing. See also 6.17.

The COSEM AL also performs some functions of the OSI presentation layer:

- encoding the ACSE APDUs in BER (ISO/IEC 8825-1). See also 7.2.3;
- encoding the xDLMS APDUs carrying the data transfer services in A-XDR (IEC 61334-6).

4.2 COSEM application layer services

4.2.1 ASO services

The services required by, or provided for the COSEM client and server APs at the logical interfaces with the respective COSEM AL, using CO procedures, fall into three categories:

- application association establishment and release;
- data transfer;
- layer management.

4.2.2 Services provided for application association establishment and release

The services provided for the establishment and release of AAs are the following:

- COSEM-OPEN;
- COSEM-RELEASE;
- COSEM-ABORT.

The COSEM-OPEN service is used to establish AAs. It is based on the ACSE A-ASSOCIATE service. It causes the start of use of an AA by those ASE procedures identified by the value of the Application_Context_Name, Security_Mechanism_Name and xDLMS context parameters. AAs may be established in different ways:

- confirmed AAs are established via a message exchange – using the COSEM-OPEN service – between the client and the server to negotiate the contexts. Confirmed AAs can be established between a single client and a single server;
- unconfirmed AAs are established via a message sent – using the COSEM-OPEN service – from the client to the server, using the parameters of the contexts supposed to be used by the server. Unconfirmed AAs can be established between a client and one or multiple servers;
- pre-established AAs may pre-exist. In this case, the COSEM-OPEN service is not used. A pre-established AA can be confirmed or unconfirmed.

The COSEM-RELEASE service is used to release AAs. If successful, it causes the completion of the use of the AA without loss of information in transit (graceful release). In some communication profiles – for example in the TCP-UDP/IP based profile – the COSEM-RELEASE service is based on the ACSE A-RELEASE service. In some other communication profiles – for example in the 3-layer, CO, HDLC-based profile – there is a one-to-one relationship between a confirmed AA and the supporting protocol layer connection. Therefore,

AAs can be released simply by disconnecting the corresponding supporting layer connection. Pre-established AAs cannot be released.

The COSEM-ABORT service causes the abnormal release of an AA with the possible loss of information in transit. It does not rely on the ACSE A-ABORT service.

The COSEM-OPEN service is specified in 6.2, the COSEM-RELEASE service in 6.3 and the COSEM-ABORT service in 6.4.

4.2.3 Services provided for data transfer

4.2.3.1 The xDLMS application service element

As mentioned in 4.1, xDLMS includes some extensions to the DLMS standard, IEC 61334-4-41. These extensions define added functionality; existing functionality is not modified. They are made in such a way, that there is no conflict with the existing DLMS standard and comprise the following:

- additional services;
- additional data types;
- new DLMS version number;
- new conformance block;
- clarification of the meaning of the PDU Size.

The additional services are GET, SET, ACTION and EventNotification. They allow to reference attributes and methods of COSEM interface objects using LN referencing. See 4.2.3.5 and 4.2.3.6.

The additional data types are specified in Clause 8.

The new DLMS version number, corresponding to the first version of the xDLMS ASE is 6.

The new xDLMS conformance block enables optimised DLMS/COSEM server implementations with extended functionality. It can be distinguished from the DLMS conformance block by its tag "Application 31". See 7.3.1 and Clause 8.

For services using SN referencing, new variants of the Variable_Access_Specification service parameter, the Read.response and the Write.response services have been added to support selective access and block transfer.

To clarify the meaning of the maximum PDU size usable by the client and the server, the modifications shown in Table 1 have been made. The xDLMS-Initiate service uses these names for PDU Sizes.

Table 1 – Clarification of the meaning of PDU Size for DLMS/COSEM

was:	new:
Page 61, Table 3, IEC 61334-4-41:1996	
Proposed Max PDU Size	Client Max Receive PDU Size
Negotiated Max PDU Size	Server Max Receive PDU Size
Page 63, 5th paragraph, IEC 61334-4-41:1996	
The Proposed Max PDU Size parameter, of type Unsigned16, proposes a maximum length expressed in bytes for the exchanged DLMS PDUs. The value proposed in an Initiate request shall be large enough to always permit the Initiate Error PDU transmission.	The Client Max Receive PDU Size parameter, of type Unsigned16, contains the maximum length expressed in bytes for a DLMS PDU that the server may send. The client will discard any received PDUs that are longer than this maximum length. The value shall be large enough to always permit the AARE APDU transmission. Values below 12 are reserved. The value 0 indicates that there is no limit on the PDU size.
Page 63, last paragraph, IEC 61334-4-41:1996	
The Negotiated Max PDU Size parameter, of type Unsigned16, contains a maximum length expressed in bytes for the exchanged DLMS PDUs. A PDU that is longer than this maximum length will be discarded. This maximum length is computed as the minimum of the Proposed Max PDU Size and the maximum PDU size than the VDE-handler may support.	The Server Max Receive PDU Size parameter, of type Unsigned16, contains the maximum length expressed in bytes for a DLMS PDU that the client may send. The server will discard any received PDUs that are longer than this maximum length. Values below 12 are reserved. The value 0 indicates that there is no limit on the PDU size.

4.2.3.2 Client/server and non-client/server type services

xDLMS data transfer services are related to attributes and methods of COSEM interface objects, specified in IEC 62056-6-2 and provide the interface between the COSEM application model and the communication protocols.

When the server uses SN referencing, attributes and methods of COSEM interface objects are mapped to DLMS named variables. The mapping is held by the Association SN interface object(s).

For accessing attributes and methods, client/server type services are used: the client requests services and the server provides them.

There is also an unsolicited, non-client/server type service. This service is requested by the server, upon occurrence of an event, to inform the client of the value of one or more attributes, as though they had been requested by the client. It is an unconfirmed service.

4.2.3.3 Referencing methods and service mapping

As specified in 4.1, there are two distinct service sets available, one for LN and one for SN referencing.

On the client side, in order to handle the different referencing methods transparently for the AP, the AL provides only one service set, using LN referencing. Using a unique, standardized service set between COSEM client APs and the communication protocol – hiding the particularities of COSEM servers using different referencing methods – allows to specify an Application Programming Interface, API. This is an explicitly specified interface corresponding to this service set for applications running in a given computing environment (for example Windows, UNIX, etc.). Using this – public – API specification, client applications can be developed without knowledge about particularities of a given server.

On the server side, services using LN referencing and/or SN referencing can be provided.

In the case of confirmed AAs, the referencing method and the service set to be used are negotiated during the AA establishment phase via the COSEM application context; see 7.2.2.2, and the conformance block; see 7.3.1. It shall not change during the lifetime of the AA established. Using LN or SN services within a given AA is exclusive.

In the case of unconfirmed and pre-established AAs, the client AL is expected to know the referencing method and the service set supported by the server.

When the server uses LN referencing, the services are the same on both side. When the server does not use LN referencing, a mapping between client side and server side services takes place. As explained in 4.1, if the server uses SN referencing this mapping is performed by the Client SN_MAPPER ASE. See also 6.17.

4.2.3.4 Confirmed and unconfirmed services

Within confirmed AAs, client/server type data transfer services can be invoked in a confirmed or unconfirmed manner. Within unconfirmed AA-s, client/server type data transfer services may be invoked in an unconfirmed manner only. See also 7.3.2.

4.2.3.5 COSEM client/server type services

COSEM client/server type data transfer services for LN referencing are the following:

- the GET service, used to read the value of one or more attributes of COSEM interface objects. See 6.6;
- the SET service, used to write the value of one or more attributes of COSEM interface objects. See 6.7;
- the ACTION service, used to invoke one or more methods of COSEM interface objects. Invoking methods may imply sending service parameters and returning data. See 6.8.

COSEM client/server type data transfer services for SN referencing are the following:

- the Read service, used to read the value of one or more attributes or to invoke one or more methods of COSEM interface objects when return parameters are expected. It is a confirmed service. See 6.10;
- the Write service, used to write the value of one or more attributes or to invoke one or more methods of COSEM interface objects when no return parameters are expected. It is a confirmed service. See 6.13;
- the UnconfirmedWrite service, used to write the value of one or more attributes or to invoke one or more methods of COSEM interface objects. It is an unconfirmed service. See 6.14.

4.2.3.6 Services for event notification

To support event notification, COSEM also provides non client/server type, unsolicited services:

- with LN referencing the EventNotification service. See 6.9;
- with SN referencing, the InformationReport service. See 6.15.

4.2.3.7 Identifying service invocations: the Invoke_Id parameter

In the client/server model, requests are sent by the client and responses are sent by the server. The client is allowed to send several requests before receiving the response to the previous ones. Therefore – to be able to identify which response corresponds to each request – it is necessary to include a reference in the request.

The `Invoke_Id` parameter is used for this purpose. The value of this parameter is assigned by the client so that each request carries a different `Invoke_Id`. The server shall copy the `Invoke_Id` into the corresponding response.

The EventNotification service – being a non-client/server type service – does not contain the `Invoke_Id` parameter.

This feature is available only with LN referencing.

4.2.3.8 Priority of service invocations: the Priority parameter

For data transfer services using LN referencing, two priority levels are available: normal (FALSE) and high (TRUE). This feature allows receiving a response to a new request before the response to a previous request is completed.

Normally, the server serves incoming service requests in the order of reception (FIFS, First In, First Served). However, a request with the priority parameter set to HIGH shall be served before the previous requests with priority NORMAL. The response shall carry the same priority flag as that of the corresponding request. Managing priority is a negotiable feature; see 7.3.1.

NOTE 1 As service invocations are identified with an `Invoke_Id`, services with the same priority can be served in any order.

NOTE 2 If the feature is not supported, requests with HIGH priority are served with NORMAL priority.

This feature is not available with services using SN referencing. The server treats the services on a FIFS basis.

4.2.3.9 Selective access

In the case of some COSEM interface classes, selective access to attributes is available, meaning that either the whole attribute or a selected portion of it can be accessed. For this purpose, access selectors and parameters are specified as part of the specification of the relevant attributes.

To use this possibility, attribute-related services can be invoked with these access selection parameters. In the case of LN referencing, this feature is called Selective access; see 6.6 and 6.7. It is a negotiable feature; see 7.3.1. In the case of SN referencing, this feature is called Parameterized access; see 0, 6.13 and 6.14. It is a negotiable feature; see 7.3.1.

4.2.3.10 Multiple references

In a data transfer service invocation, it is possible to reference one or several attributes respectively methods. Using multiple references is a negotiable feature. See 7.3.1.

4.2.3.11 Attribute_0 referencing

With the GET and SET services, a special feature, `Attribute_0` referencing, is available. By convention, attributes of COSEM interface objects are numbered from 1 to n, where `Attribute_1` is the logical name of the COSEM interface object. `Attribute_0` has a special meaning: it references all attributes with positive index (public attributes). The use of `Attribute_0` referencing with the GET service is explained in 6.6 and with the SET service in 6.7. `Attribute_0` referencing is a negotiable feature; see 7.3.1.

NOTE As specified in IEC 62056-6-2:—, 4.1 and 4.2, manufacturers can add proprietary methods and/or attributes to any object, using negative numbers.

4.2.3.12 Transferring long service parameters

The data transfer services are carried by the APDUs, exchanged between the peer ALs, in an encoded form. In some cases, the data may be longer than the Client / Server Max Receive PDU Size negotiated. To transfer such 'long' data, two transporting mechanisms are available:

- long data transfer using the AL block transfer protocol. This mechanism can be used with any confirmed client/server type services. It is a negotiable feature. See 7.3.1;
- long data transfer in a transparent manner to the AL. This feature can be used when the supporting layer(s) provide(s) segmentation; see Annex A.

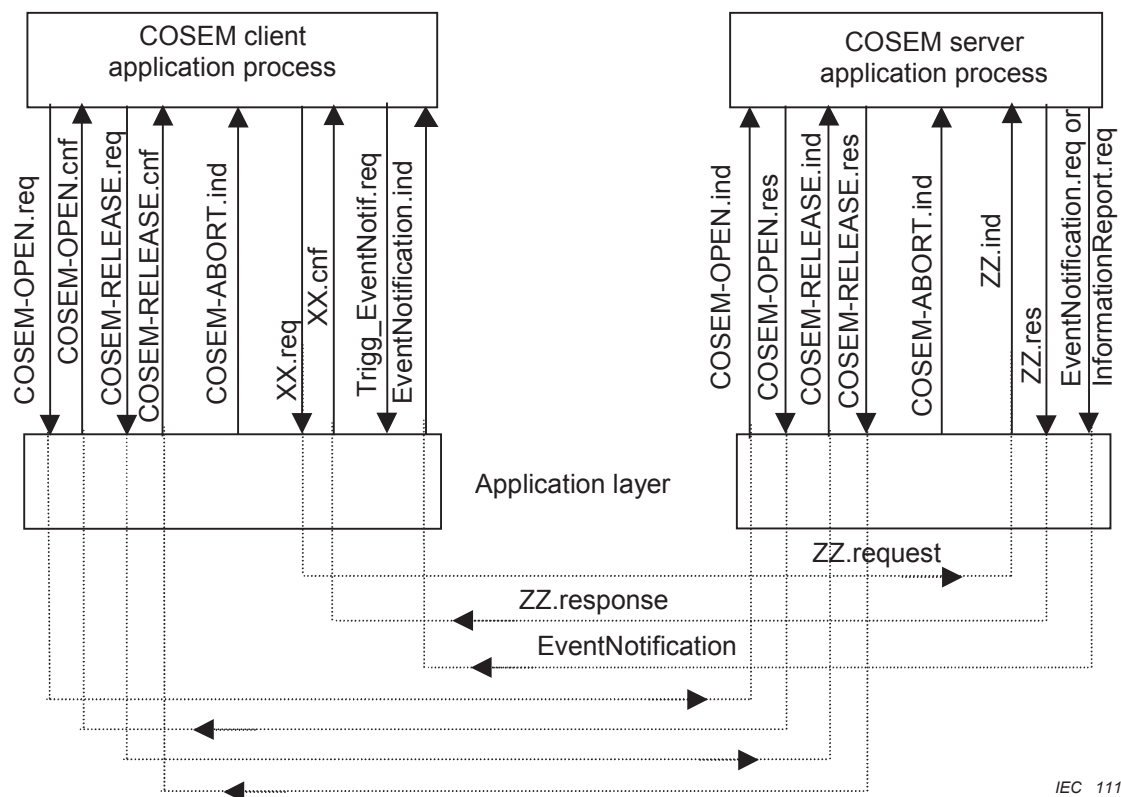
NOTE There is no mechanism within AL block transfer for re-sending or requesting a missing or corrupt block. Such a mechanism can be a feature of long data transfer managed transparently by the supporting layers.

4.2.4 Layer management services

Layer management services have local importance only. Therefore, specification of these services is not within the scope of this companion specification. The specific SetMapperTable service is defined in 6.16.

4.2.5 Summary of COSEM application layer services

A summary of the services available at the top of the COSEM AL is shown in Figure 2. Layer management services are not shown. Although the service primitives are different on the client and server side, the APDUs are the same. The abstract syntax of the ACSE and COSEM APDUs is specified in Clause 8.



IEC 1113/13

NOTE XX and ZZ refer to client/server type data transfer services. These services are different on the client side and the server side, if the server does not use LN referencing. See 6.17.

Figure 2 – Summary of COSEM AL services

4.3 COSEM application layer protocols

The COSEM AL protocols specify the procedures for information transfer for AA control and authentication using connection-oriented ACSE procedures, and for data transfer between COSEM clients and servers using xDLMS procedures. Therefore, the AL protocol is based on the ACSE and the DLMS protocols, as specified in ISO/IEC 15954 and in IEC 61334-4-41 – with the extensions for DLMS/COSEM – respectively. The procedures are defined in terms of:

- the interactions between peer ACSE and xDLMS protocol machines through the use of services of the supporting protocol layer;
- the interactions between the ACSE and xDLMS protocol machines and their service user;
- the abstract syntax of the APDUs using ASN.1 as specified in ISO/IEC 8824-1.

The COSEM AL protocols are specified in Clause 7.

5 Information security in DLMS/COSEM

NOTE Clause 5 is based on parts of NIST documents. Reprinted courtesy of the National Institute of Standards and Technology, Technology Administration, U.S. Department of Commerce. Not copyrightable in the United States.

5.1 Definitions

5.1.1

confidentiality

preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information

[SOURCE: FIPS PUB 199]

5.1.2

integrity

guarding against improper information modification or destruction, and includes ensuring information non-repudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information

[SOURCE: FIPS PUB 199]

5.2 General

Confidentiality and integrity are among the key requirements when open systems connect to public media. With the increase of computational power, the requirements for strong cryptographic methods also increase. DLMS/COSEM provides two main information security features for accessing and transporting data:

- *data access security* controls access to the data held by a DLMS/COSEM server. See 5.3;
- *data transport security* allows the sending party to apply cryptographic protection to the xDLMS APDUs sent to ensure confidentiality and integrity. This requires ciphered ADPUs. The receiving party can remove or check this protection. See 5.4.

These information security features are provided partly by the COSEM AL, partly by COSEM objects:

- upon AA establishment, two contexts are negotiated using the ACSE services: the *application context* and the *authentication context*. The application context determines whether ciphered APDUs can be used or not. The authentication context determines the level of data access security. The ACSE APDUs are not cryptographically protected;

The user-information field of the ACSE APDUs shall be cryptographically protected as determined by the application-context and the security policy.

- once an AA is successfully established, COSEM services can be used to access attributes and methods of COSEM objects, depending on the access rights valid in the given association. These services are carried by xDLMS APDUs, which may be cryptographically protected: authenticated, encrypted, or both, depending on the security policy in force. The cryptographic protection is applied, removed or checked by the COSEM AL. The COSEM AP of the sending party informs, via service parameters, the COSEM AL about the protection to be applied to the APDU to be sent. The COSEM AL of the receiving party informs the COSEM AP about the protection that has been applied to the APDU received.

The message security methods described in this part of IEC 62056 have been selected from the standards specified by NIST and the IETF.

For an overview of cryptography; see Annex E.

5.3 Data access security

5.3.1 Overview

Data access security concerns role based access to data in a DLMS/COSEM device.

It is managed by the Association LN / Association SN objects. Each COSEM server i.e. a logical device may support AAs with various clients, each having a different role, and with this, different access rights. Each AA is identified with a pair of lower layer addresses. Each Association object provides a list of objects visible in that particular AA and the access rights to their attributes and methods.

To be able to access data, the client shall be properly authenticated. Upon AA establishment, an authentication context is negotiated between the client and the server. This specifies the required authentication of the peers, and, where needed, the security algorithm to verify the authentication. Three data access security levels are provided:

- Lowest level security (no security);
- Low Level Security (LLS);
- High Level Security (HLS).

5.3.2 Lowest level security (no security)

The purpose of lowest level security is to allow the client to retrieve some basic information. This authentication context does not require any peer authentication; it allows direct access to the data in the server, within the access rights available in the given AA.

5.3.3 Low Level Security (LLS)

The purpose of Low Level Security is to allow the authentication of clients by verifying the password supplied. The server is not authenticated. This authentication context is typically used when the communication channel offers adequate security to avoid eavesdropping and message (password) replay.

The client has to supply the correct password during the process of AA establishment. If the password is OK, the AA is established and the client can access data within the access rights available in the given AA. Otherwise, the AA is not established.

The LLS authentication process is supported by the COSEM-OPEN service of the COSEM AL – see 6.2 – and it is as follows:

- the client transmits a “secret” (for example a password) to the server, by using the “Calling_Authentication_Value” parameter of the COSEM-OPEN.request service primitive;
- the server checks if the “secret” received corresponds to the client identification;
- if yes, the client is authenticated and the AA can be established. If no, the AA is rejected;
- the result with diagnostic information is sent back by the server using the COSEM-OPEN.response service primitive.

The association objects provide a method/attribute to change the “secret”; see IEC 62056-6-2:—, 5.3.1, 5.3.2.

The authentication process with LLS is shown on the left side of Figure 3.

5.3.4 High Level Security (HLS)

The purpose of High Level Security is to allow mutual authentication of the client and the server participating in an association. This authentication context is typically used when the communication channel offers no intrinsic security and precautions have to be taken against eavesdroppers and against message (password) replay.

HLS requires that both the client and the server mutually authenticate each other. This is a 4-pass process, involving the exchange of challenges during AA establishment, which is followed by exchanging the results of processing these challenges, using cryptographic methods. If the authentication takes place, the client can proceed to access data within the access rights available in the given AA, and it accepts data coming from the server. Otherwise, the AA is not established.

The HLS authentication process is supported by the COSEM-OPEN service – see 6.2 – and by the Association LN / SN objects as follows:

Pass 1: The client transmits a “challenge” CtoS – for example, a random string – to the server;

Pass 2: The server transmits a “challenge” StoC – for example, a random string – to the client;

The length of the challenges shall be 8 octets to 64 octets.

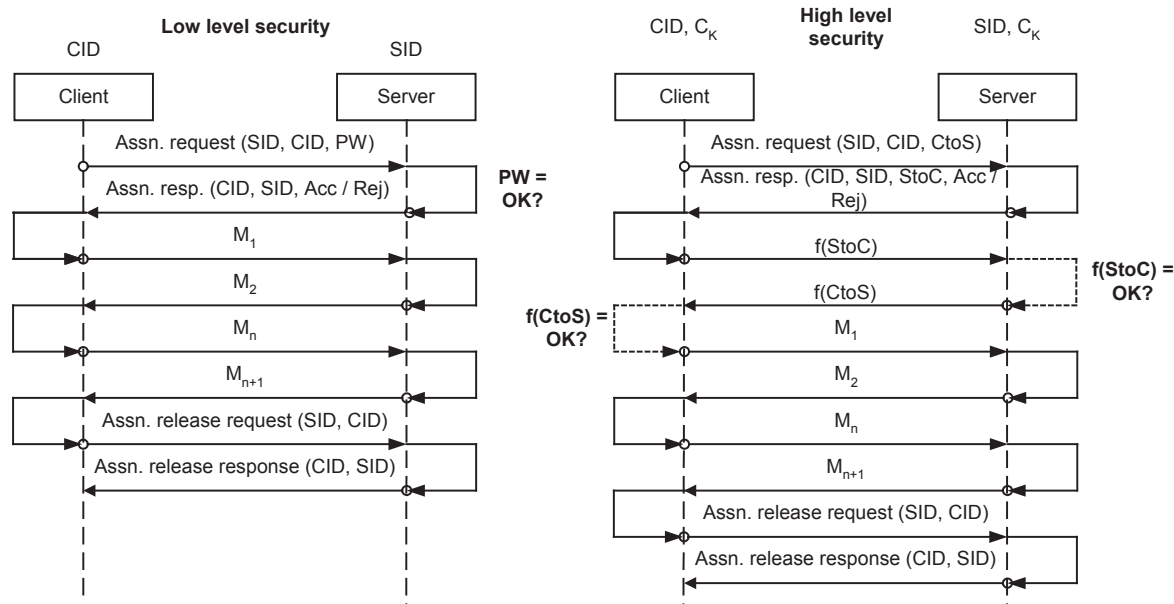
Pass 3: The client processes a “challenge” StoC according to the rules of the HLS authentication mechanism negotiated:

- in the case of HLS authentication_mechanism_id(2), the method of processing the challenge is secret – for example encrypting with a secret key – which is the HLS secret known by both the client and the server;
- in the case of HLS authentication_mechanism_id(3), the client right concatenates StoC received during Pass2 with the HLS secret, known by both the client and the server, and generates a digest using the MD5 algorithm as specified in RFC 1321;
- in the case of HLS authentication_mechanism_id(4), the process is the same, but the client generates a digest using the SHA-1 algorithm. FIPS PUB 180-1 applies in this respect;
- in the case of HLS authentication_mechanism_id(5), an authentication tag is calculated using GMAC. See 5.4.8.4.

The result – f(StoC) – is sent back to the server. The server checks if f(StoC) is the result of correct processing and – if so – it accepts the authentication of the client.

Pass 4: If the client is authenticated, the server processes the “challenge” CtoS in the same way as described in Pass3. The result – f(CtoS) – is sent back to the client. The client checks if f(CtoS) is the result of correct processing and – if so – it accepts the authentication of the server.

The authentication process with HLS is shown on the right side of Figure 3.



IEC 1114/13

Figure 3 – LLS and HLS authentication

Pass 1 is supported by the COSEM-OPEN.request service primitive of the client AL. The parameter "Security_Mechanism_Name" carries the identifier of the HLS mechanism, and the parameter "Calling_Authentication_Value" carries the challenge CtoS.

Pass 2 is supported by the COSEM-OPEN.response service primitive of the server AL. The parameter "Security_Mechanism_Name" carries the identifier of the HLS mechanism, and the parameter "Responding_Authentication_Value" carries the challenge StoC.

After Pass 2, the AA is formally established, but the access of the client is restricted to the method "reply_to_HLS_authentication" of the current "Association" object.

Pass 3 and Pass 4 are supported by the method reply_to_HLS_authentication of the association object(s); see IEC 62056-6-2:—, 5.3.1 and 5.3.2. If both passes are successfully executed, then full access is granted according to the current AA. Otherwise, either the client or the server aborts the AA.

In addition, the Association SN / LN object provides the method to change the HLS "secret": change_HLS_secret.

REMARK After the client has issued the change_HLS_secret () – or change_LLS_secret () – method, it expects a response from the server acknowledging that the secret has been changed. It is possible that the server transmits the acknowledgement, but due to communication problems, the acknowledgement is not received at the client side. Therefore, the client does not know if the secret has been changed or not. For simplicity reasons, the server does not offer any special support for this case; i.e. it is left to the client to cope with this situation.

5.4 Data transport security

5.4.1 Applying, removing or checking the protection: ciphering and deciphering

For data transport security, cryptographic protection can be applied before sending an xDLMS APDU – as determined by the security policy; see 5.4.3 – then removed or checked after the reception of an APDU.

NOTE Cryptographic protection of data in storage is out of the scope of this companion specification.

Applying the protection is achieved by ciphering. Deciphering removes or checks the protection, restoring the original xDLMS APDU. Ciphering and deciphering is performed by the COSEM AL as shown in Figure 4.

When a COSEM service request or response primitive is invoked by the client or the server AP respectively, the service parameters include the Security_Options parameter. This parameter informs the AL on the requested security to be applied on the xDLMS APDU carrying the service primitive, and it may identify elements of the security material to be used.

Similarly, a COSEM indication or confirm service primitive includes a Security_Status parameter. This parameter informs the AP on the security that was applied on the xDLMS APDU carrying the service primitive, and it may include information on the elements of the security materials used. The authentication tag – if any – is also passed.

For COSEM service specifications, see Clause 6.

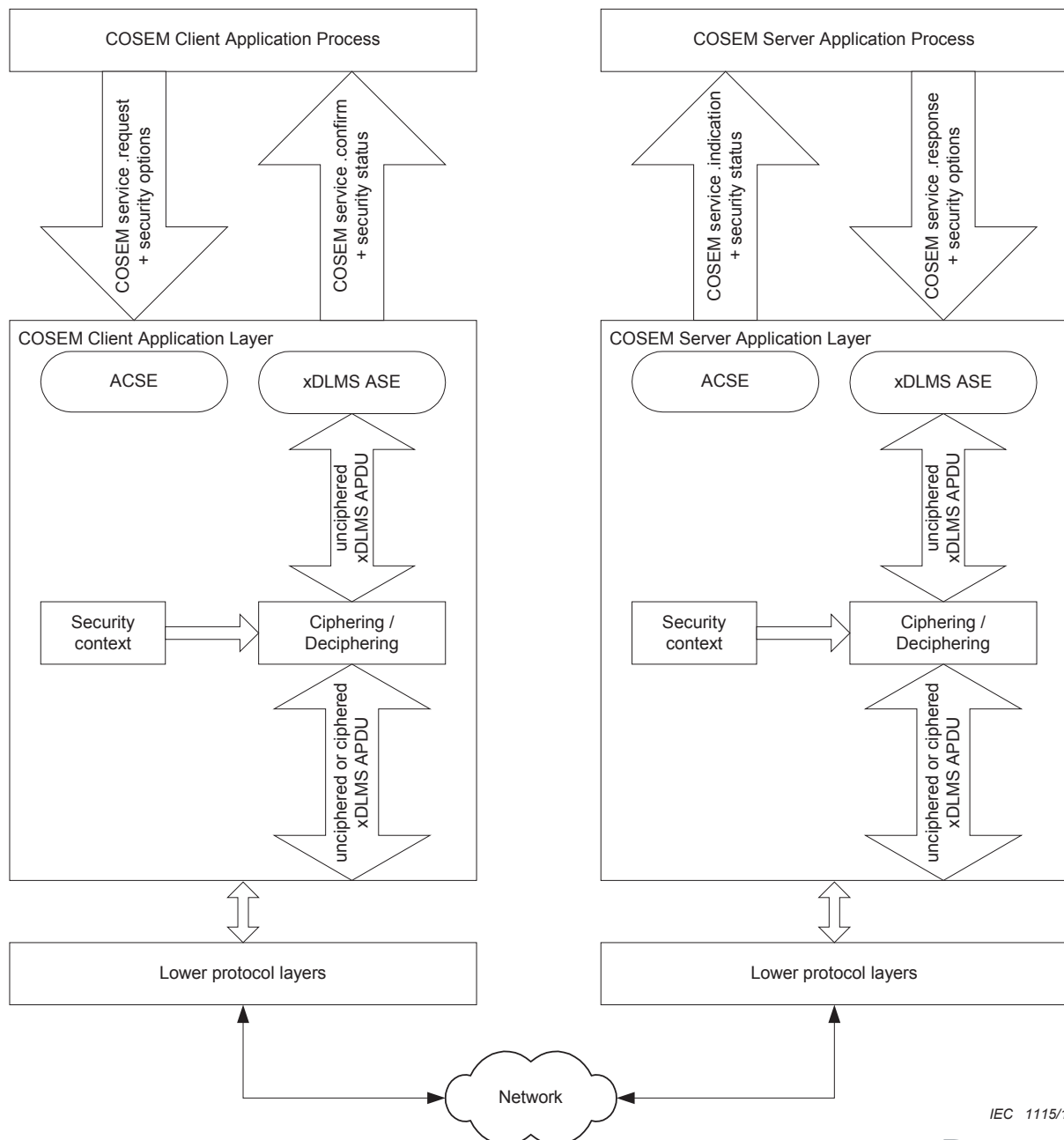


Figure 4 – Data transport security in DLMS/COSEM

5.4.2 Security context

The security context defines security attributes relevant for the ciphering / deciphering process:

- the security policy in force, determining the kind of protection to be applied. See 5.4.3;
- the security suite, specifying the security algorithm(s). See 5.4.4;
- the security material relevant for the given security suite, including elements like block cipher keys, authentication keys, initialization vectors etc. See 0.

5.4.3 Security policy

The following security policies are specified:

- security is not imposed;
- all messages are authenticated;
- all messages are encrypted;
- all messages are authenticated and encrypted.

The security policy is held by the `security_policy` attribute of the Security setup object; see IEC 62056-6-2:—, 5.3.5.

Authenticated access may be required selectively: certain attributes and methods may be accessible only by using authenticated messages. This is indicated by the access mode to each attribute and method; see IEC 62056-6-2:—, 5.3.1 and 5.3.2 respectively. Therefore, authenticated xDLMS APDUs may be used – within a ciphered application context – even when the security policy in effect does not require that all messages be authenticated.

Note that if authenticated access is required, but the application context is not a ciphered one, the attributes and methods requiring authentication cannot be accessed. If the client attempts to send a ciphered APDU in an unciphered application context, then the server application layer shall reject it. In the case of SN referencing, the response shall be `ConfirmedServiceError` APDU, carrying the tag of the rejected APDU. In the case of LN referencing, the response shall be an `ExceptionResponse` APDU.

On the other hand, encryption applies generally for all messages within a given AA: if the security policy requires that all messages shall be encrypted, then all APDUs shall be encrypted – and additionally they may be authenticated.

When the security policy requires that all messages shall be authenticated and encrypted, then only APDUs that are both encrypted and authenticated can be used.

Messages protected by higher security than what the security policy requires are always allowed (provided that the application context negotiated allows them).

5.4.4 Security suite

A security suite determines the cryptographic algorithm(s) used for message security. A security suite is identified with a Security Suite ID; see Table 2. Currently, one security suite is specified, the Galois/Counter Mode (GCM) with AES-128. In this security suite, global keys – see 5.4.7.3.3 – are protected during transportation using the AES-128 key wrap algorithm.

Table 2 – Security suites

Security Suite Id	Authentication algorithm	Encryption algorithm	Key transport method
0	AES-GCM-128	AES-GCM-128	Key wrapping using AES-128 key wrap
All other reserved	–	–	–

5.4.5 Security material

The elements of the security material are the following:

- a block cipher key, denoted EK ;
- an authentication key, denoted AK ;
- an initialization vector, denoted IV .

The elements depend on the security suite. For AES-GCM-128, they are defined in 5.4.8.

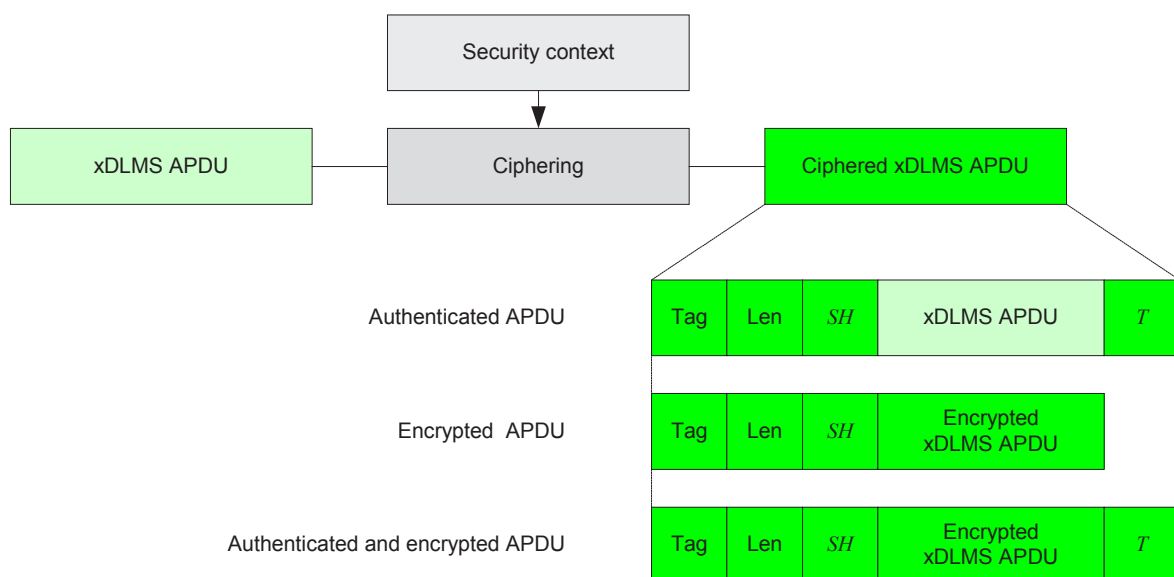
5.4.6 Ciphersed xDLMS APDUs

Ciphersed xDLMS APDUs can be used in a ciphersed COSEM application context only. A ciphersed APDU is always an octet string.

In a ciphersed COSEM application context, unciphersed APDUs may also be used.

The structure of the ciphersed APDU depends on the kind of ciphering applied; see Figure 5:

- if the APDU is authenticated only, it contains the APDU tag, the length field, the security header field, the original APDU and the authentication tag;
- if the APDU is encrypted only, it contains the APDU tag, the length field, the security header field, and the encrypted DLMS APDU;
- if the APDU is both authenticated and encrypted, it contains the APDU tag, the length field, the security header field, the encrypted APDU and the authentication tag.



IEC 1116/13

Figure 5 – Ciphersed xDLMS APDUs

The APDU tag identifies the type of the APDU; see Clause 8.

The length field specifies the length of the octet-string (including the security header, the unencrypted or encrypted xDLMS APDU and the authentication tag).

The Security Header, *SH*, includes the Security Control field, *SC* right concatenated with the Frame Counter, *FC*.

The Security Control field is shown in Table 3, where:

- Bit 3...0: Security_Suite_Id; see 5.4.4;
- Bit 4: “A” subfield: indicates that the APDU is authenticated;
- Bit 5: “E” subfield: indicates that the APDU is encrypted;
- Bit 6: Key_set subfield 0 = Unicast;
1 = Broadcast; see 5.4.7.2;
- Bit 7: Reserved, shall be set to 0.

Table 3 – Security control field

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3...0
Reserved, set to 0	Key_set	E	A	Security_Suite_Id

The Frame Counter is an internal counter maintained by the sending and receiving parties.

Finally, *T* is the authentication tag, calculated by the authentication algorithm.

5.4.7 Cryptographic keys

5.4.7.1 General

A cryptographic key is a parameter used in conjunction with a cryptographic algorithm that determines its operation in such a way that an entity with knowledge of the key can reproduce or reverse the operation, while an entity without knowledge of the key cannot. For the purposes of DLMS/COSEM, examples of operations include:

- the transformation of plaintext data into ciphertext data;
- the transformation of ciphertext data into plaintext data;
- the computation of an authentication code from data;
- the verification of an authentication code from data and a received authentication code.

5.4.7.2 Key types

Various key types are defined. They are identified according to their classification, their cryptographic function, their use and their scope.

By their classification, a distinction is made between:

- *symmetric* keys. A symmetric key is a single cryptographic key that is used with a secret (symmetric) key algorithm. For an overview of symmetric key algorithms; see E.3;
- *public* and *private* keys, used with public key cryptographic algorithms. For an overview of asymmetric key algorithms; see E.4.

For symmetric keys, a distinction is made between:

- *Symmetric authentication keys*, used with symmetric key algorithms to provide assurance of the integrity and source of messages;
- *Symmetric data encryption keys*, used with symmetric key algorithms to apply confidentiality protection to information; and

- *Symmetric key wrapping keys*, used to encrypt other keys using symmetric key algorithms. Key wrapping keys are also known as key encrypting keys (KEK).

By their use, a distinction is made between:

- *unicast keys*, used to secure unicast communication between two peers;
- *broadcast keys*, used to secure broadcast communication between a DLMS/COSEM client and several DLMS/COSEM servers. In DLMS/COSEM, broadcast can be initiated by the client only.

By their scope, a distinction is made between *dedicated keys* and *global keys*. These terms are DLMS/COSEM specific:

- a *dedicated key* is a ciphering key that may be used to cipher xDLMS APDUs from the establishment of an AA between a client and a server until the AA is released. Therefore, its lifetime is the same as the lifetime of the AA. It is delivered during AA establishment and used in subsequent transmissions within the same AA.

When the AA is released, the dedicated key should be destroyed by the server as well as by the client.

- a *global key* is a ciphering key that may be (re-)used to cipher xDLMS APDUs, exchanged between the same client and server, in more than one instance of the AA.

The following (symmetric) encryption keys may be available: global unicast key, global broadcast key, dedicated unicast key.

A (symmetric) authentication key is always a global key that can be used in unicast and broadcast messages.

In Addition, security suite specific rules may apply. See also Table 4.

5.4.7.3 Key management

5.4.7.3.1 System architectures

NOTE Subclause 5.4.7.3 provides general guidelines for secure implementation. The management of the security system, including the security keys is the responsibility of the system operator.

For the purposes of 5.4.7.3, two general system architectures are considered:

- the COSEM client is the central DCS, exchanging data directly with COSEM servers;
- the COSEM client is a concentrator, exchanging data on the one hand with COSEM servers, and on the other hand with the central DCS. In this case, the concentrator may or may not be a COSEM server at the same time. The central DCS may or may not be a COSEM client.

For generation and distribution of symmetric keys, NIST SP 800-57:2006, 8.1.5.2 applies.

5.4.7.3.2 Master key

A master key shall be present in each COSEM server logical device configured in the system. This key is used for wrapping global keys; see 5.4.7.3.3.

NOTE 1 The way of generating and installing the master key is out of the scope of this companion specification.

A secure mechanism shall be in place to ensure that the central data DCS knows the master key of each COSEM server logical device participating in the system.

NOTE 2 The specification of such a mechanism is out of the scope of this companion specification.

The concentrators, if they are present in the system, shall not know the master keys

For security reasons, the lifetime of the master key should be limited. The way to replace the master keys is out of the scope of this standard. For guidance, see NIST SP 800-57:2006, 8.2.3, Key change function.

5.4.7.3.3 Global keys

Before ciphering can be used, global keys have to be generated and delivered to the COSEM logical devices participating in the system.

Global keys shall be generated by the central DCS.

The way to generate the global keys is outside the scope of this companion specification. For security reasons, their lifetime should be limited.

For delivery, global keys shall be wrapped using the AES-128 key wrap algorithm as specified in RFC 3394, and using the master key as the Key Encryption Key (KEK).

When the system works without (a) concentrator(s), the global keys shall be delivered by the central DCS invoking the `global_key_transfer` method of the Security setup interface object in the COSEM server with the wrapped global key as a parameter; see IEC 62056-6-2:—, 5.3.5.

When the system works with (a) concentrator(s), the global keys shall be delivered – wrapped by the master key – to the concentrator. The `global_key_transfer` method of the Security setup interface object in the COSEM server shall be invoked then by the concentrator with the wrapped global key as a parameter. For details, see IEC 62056-6-2:—, 5.3.5.

In any case, the global key is wrapped by the master key.

The concentrator should not know the master keys, therefore it should not be able to wrap / unwrap global keys.

The global keys shall also be made available by the central DCS to the concentrator(s), if present, in a secure way.

NOTE The specification of such a process is out of the scope of this part of IEC 62056.

5.4.7.3.4 Dedicated keys

When the system works without (a) concentrator(s), dedicated keys shall be generated by the central DCS. When the system works with (a) concentrator(s), dedicated keys shall be generated by the concentrators.

For delivery, they shall be included in the dedicated-key field of the xDLMS InitiateRequest APDU, carried by the user-information field of the AARQ APDU. The xDLMS InitiateRequest APDU shall be authenticated and encrypted using the AES-GCM-128 algorithm, the global unicast encryption key and – if in use, see 5.4.8.3.4.4, – the authentication key. Likewise, the xDLMS InitiateResponse APDU, carried by the user-information field of the AARE APDU shall be also encrypted and authenticated the same way. See also 5.2.

The various cryptographic key types, together with their use and their management are summarized in Table 4.

Table 4 – Cryptographic keys and their management

Key type	Use	Generation	Delivery	Location
Symmetric keys				
Master key	Key Encryption Key (KEK) for global keys	Not specified	Not specified	Central DCS and COSEM server
Global unicast encryption key	Global ciphering of unicast xDLMS APDUs	Central DCS	Wrapped with master key, invocation of global_key_transfer method by the central DCS or the concentrator	Central DCS (and Concentrator) and COSEM server
Global broadcast encryption key	Global ciphering of broadcast xDLMS APDUs	Central DCS	Wrapped with master key, invocation of global_key_transfer method by the central DCS or the concentrator	Central DCS (and Concentrator) and COSEM server
Authentication key (Global)	Authentication of xDLMS APDUs	Central DCS	Wrapped with master key, invocation of global_key_transfer method by the central DCS or the concentrator	Central DCS (and Concentrator) and COSEM server
Dedicated (unicast) encryption key	Dedicated ciphering of unicast xDLMS APDUs	Central DCS or Concentrator	Transported as part of the xDLMS InitiateRequest APDU, which is encrypted and authenticated using the AES-GCM-128 algorithm, the global unicast encryption key and the authentication key	Central DCS (or Concentrator) and COSEM server (during the lifetime of the AA)
NOTE If alternatives are given, the first alternative is valid for systems not using concentrators and the second alternative for systems using concentrators.				

5.4.8 The Galois/Counter Mode of Operation (GCM)

5.4.8.1 General

NOTE The following text is taken from NIST SP 800-38D:2007, Clause 3.

Galois/Counter Mode (GCM) is an algorithm for authenticated encryption with associated data. GCM is constructed from an approved symmetric key block cipher with a block size of 128 bits, such as the Advanced Encryption Standard (AES) algorithm; see FIPS PUB 197. Thus, GCM is a mode of operation of the AES algorithm.

GCM provides assurance of the confidentiality of data using a variation of the Counter mode of operation for encryption.

GCM provides assurance of the authenticity of the confidential data (up to about 64 gigabytes per invocation) using a universal hash function that is defined over a binary Galois (i.e., finite) field. GCM can also provide authentication assurance for additional data (of practically unlimited length per invocation) that is not encrypted.

If the GCM input is restricted to data that is not to be encrypted, the resulting specialization of GCM, called GMAC, is simply an authentication mode on the input data.

GCM provides stronger authentication assurance than a (non-cryptographic) checksum or error detecting code; in particular, GCM can detect both accidental modifications of the data and intentional, unauthorized modifications.

In DLMS/COSEM, it is also possible to use GCM to provide confidentiality only: in this case, the authentication tags are simply not computed and checked.

For more information, see NIST SP 800-38D.

5.4.8.2 Definitions, abbreviations, symbols and notation relevant for the Galois/Counter mode

Term / Abbreviation	Symbol	Meaning
Additional authenticated data (AAD)	A	Input data to the authenticated encryption function that is authenticated but not encrypted. It is also known as Associated Data.
Authenticated decryption		Function of GCM in which the ciphertext is decrypted into the plaintext, and the authenticity of the ciphertext and the AAD are verified.
Authenticated encryption		Function of GCM in which the plaintext is encrypted into the ciphertext, and an authentication tag is generated on the AAD and the ciphertext.
Authentication key	AK	Part of the AAD
Block cipher		Parameterized family of permutations on bit strings of a fixed length; the parameter that determines the permutation is a bit string called the key.
Ciphertext	C	Encrypted form of the plaintext
	EK	Block cipher key
Fixed field		In the deterministic construction of IVs, field that identifies the device or context for the instance of the authenticated encryption function.
	FC	Frame counter.
Fresh		For a newly generated key, property of being unequal to any previously used key.
GCM		Galois/Counter Mode
Initialization vector	IV	Nonce that is associated with an invocation of authenticated encryption on a particular plaintext and AAD.
Invocation field		In the deterministic construction of IVs, field that identifies the sets of inputs to the authenticated encryption function in a particular device or context. For the purposes of this companion specification, the invocation field is the frame counter.
Key		Parameter of the block cipher that determines the selection of the forward cipher function from the family of permutations.
	KEK	Key encryption key
	$len(X)$	Bit length of the bit string X
	$LEN(X)$	Octet length of the octet string X
Nonce		Value that is used only once within a specified context.
Plaintext	P	Input data to the authenticated encryption function that is both authenticated and encrypted.
	SC	Security control byte
	SH	Security header; concatenation of the security control byte SC and the frame counter FC : $SH = SC \parallel FC$.
System title	$Sys-T$	A unique identifier of the system.
Authentication tag	T	Cryptographic checksum on data that is designed to reveal both the accidental errors and the intentional modification of the data.
	t	Bit length of the authentication tag NOTE This is the same as $len(T)$.
	$X \parallel Y$	Concatenation of two strings, X and Y .

5.4.8.3 Elements of GCM

5.4.8.3.1 Block cipher and keys

For the purposes of this companion specification, the AES encryption standard has been selected. GCM uses the forward cipher function; it does not employ the inverse cipher function.

GCM uses a single key, the block cipher key (the key, for short). Its size shall be 128 bits. In DLMS/COSEM, for additional security, an authentication key is also specified. It is part of the additional authenticated data, AAD. The block cipher key is denoted *EK* and the authentication key is denoted *AK*.

5.4.8.3.2 GCM functions

5.4.8.3.2.1 General

NOTE The following is based on NIST SP 800-38D:2007, 5.2.

The two functions that comprise GCM are called authenticated encryption and authenticated decryption. The authenticated encryption function encrypts the confidential data and computes an authentication tag on both the confidential data and any additional, non-confidential data. The authenticated decryption function decrypts the confidential data, contingent on the verification of the tag.

When the input is restricted to non-confidential data, the resulting variant of GCM is called GMAC. For GMAC, the authenticated encryption and decryption functions become the functions for generating and verifying an authentication tag on the non-confidential data.

Finally, if authentication is not required, the authenticated encryption function encrypts the confidential data, but the authentication tag is not computed. The authenticated decryption function decrypts the confidential data, but no authentication tag is computed and verified.

5.4.8.3.2.2 Authenticated encryption – input and output data

NOTE The following text is based on NIST SP 800-38D:2007, 5.2.1.1 and RFC 4106:2005, 2.1.

The authenticated encryption operation has four inputs, each of which shall be an octet string:

- a plaintext, denoted *P*;
- additional authenticated data (AAD), denoted *A*. It shall include – together with other elements – the authentication key, denoted *AK*.
- a secret block cipher key denoted *EK*;
- an initialization vector denoted *IV*.

The plaintext and the AAD are the two categories of data that GCM protects. GCM protects the authenticity of the plaintext and the AAD; GCM also protects the confidentiality of the plaintext, while the AAD is left in the clear.

The IV is essentially a nonce, i.e. a value that is unique within the specified (security) context, which determines an invocation of the authenticated encryption function on the input data to be protected. See 5.4.8.3.4.5.

There are two outputs:

- a ciphertext, denoted *C* whose length is exactly that of the plaintext *P*;
- an authentication tag, denoted *T*.

5.4.8.3.2.3 Authenticated decryption – input and output data

The authenticated decryption operation has five inputs:

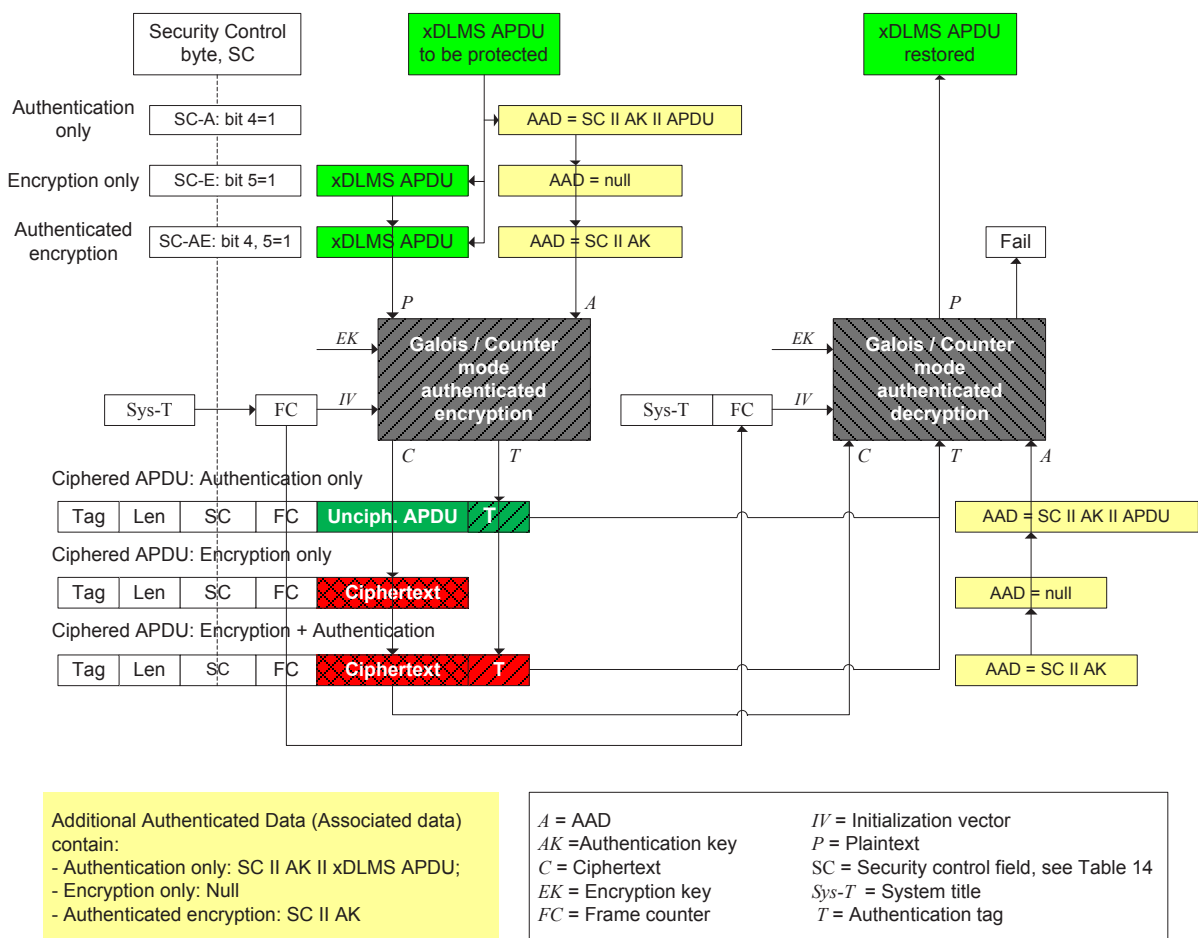
- the secret block cipher key, denoted EK ;
- the initialization vector, denoted IV ;
- the ciphertext, denoted C ;
- the additional authenticated data (AAD), denoted A ;
- the authentication tag, denoted T .

The output is one of the following:

- the plaintext P that corresponds to the ciphertext C , or
- a special error code, denoted *FAIL* in this document.

The output P indicates that T is the correct authentication tag for IV , A , and C ; otherwise, the output is *FAIL*.

The authenticated encryption and authenticated decryption functions, their inputs and outputs, as well as the structure of the secured APDUs are shown in Figure 6 below.



IEC 1117/13

Figure 6 – Cryptographic protection of xDLMS APDUs using GCM

5.4.8.3.3 Security header with GCM

The security header shall be as specified in 5.4.6, with *security_suite_id* = 0 and with the appropriate values of the *Key_set*, *E* and *A* sub-fields.

5.4.8.3.4 GCM parameters and data elements

5.4.8.3.4.1 Length of the authentication tag

The bit length of the authentication tag, denoted *t*, is a security parameter. Its value shall be 96 bits.

5.4.8.3.4.2 Plaintext and additional authenticated data

The plaintext, denoted *P* and the additional authenticated data, denoted *A* depend on the kind of protection. They are shown in Table 5, where *SC* is the security control byte, *AK* is the authentication key and *M* is the message, i.e. the xDLMS APDU to be protected.

The bit lengths of *P* and *A* shall meet the following requirements:

- $\text{len}(P) < 2^{39}-256$;
- $\text{len}(A) < 2^{64}-1$;
- the bit lengths of *P* and *A* shall all be multiples of 8, so that these values are octet strings.

Table 5 – Plaintext and additional authenticated data

Security control, <i>SC</i>		Security	<i>P</i>	<i>A</i>
E field	A field			
0	0	None	–	–
0	1	Authenticated only	–	<i>SC</i> <i>AK</i> <i>M</i>
1	0	Encrypted only	<i>M</i>	–
1	1	Encrypted and authenticated	<i>M</i>	<i>SC</i> <i>AK</i>

5.4.8.3.4.3 Block cipher key, *EK*

The size of the block cipher key, denoted *EK* shall be 128 bits (16 octets): $\text{len}(EK) = 128$.

The key shall be generated uniformly at random or close to uniformly at random i.e., so that each possible key is (nearly) equally likely to be generated. Consequently, the key will be fresh, i.e., unequal to any previous key, with high probability. The key shall be secret and shall be used exclusively for GCM with the chosen block cipher AES. Additional requirements on the establishment and management of keys are discussed in NIST SP 800-38D:2007, 8.1.

5.4.8.3.4.4 Authentication key, *AK*

Although not required by GCM, an authentication key denoted *AK* is specified for additional security in DLMS/COSEM. For its length and its generation, the same rules apply as for the encryption key. The authentication key shall be part of the additional authenticated data AAD. If an authentication key has been transferred, it shall be used, until a null authentication key is transferred.

5.4.8.3.4.5 Initialization vector, *IV*

For the construction of the initialization vector, *IV*, deterministic construction as specified in NIST SP 800-38D 8.2.1 shall be used.

In the deterministic construction, the IV is the concatenation of two fields, called the fixed field and the invocation field. The fixed field shall identify the physical device, or, more generally, the (security) context for the instance of the authenticated encryption function. The invocation field shall identify the sets of inputs to the authenticated encryption function in that particular device.

For any given key, two distinct physical devices shall not share the same fixed field, and two distinct sets of inputs to any single device shall not share the same invocation field.

The length of the IV shall be 96 bits (12 octets): $len(IV) = 96$. Within this:

- the leading (i.e. the leftmost) 64 bits (8 octets) shall hold the fixed field. It shall contain the system title; see 5.4.8.3.4.6;
- the trailing (i.e. the rightmost) 32 bits shall hold the invocation field; see 5.4.8.3.4.7.

The bit length of the fixed field limits the number of distinct physical devices that can implement the authenticated encryption function for the given key to 2^{64} . The bit length of the invocation field limits the number of invocations of the authenticated encryption function to 2^{32} with any given input sets without violating the uniqueness requirement.

5.4.8.3.4.6 System title

Each physical device in the system, including the meters, the concentrators when used, and the DCS shall be uniquely identified with its system title. The system title denoted as *Sys-T* shall be constructed as follows:

- its length shall be 8 octets;
- the leading (i.e., the 3 leftmost) octets shall hold the three-letter manufacturer ID⁶. This is the same as used in the leading octets of the Logical Device Name of server logical devices;

The three-letter manufacturer ID shall be used to identify all client and server systems.

NOTE 1 It can be obtained from the DLMS UA.

- the trailing (i.e. the 5 rightmost) octets, together with the manufacturer ID, shall ensure the uniqueness of the system title.

NOTE 2 These 5 octets can be derived for example from the last twelve digits of the manufacturing number, up to 999 999 999 999. This value converts to 0xE8D4A50FFF. Values above this, up to 0xFFFFFFFF (decimal value 1 099 511 627 775) can also be used, but these values cannot be mapped to the last 12 digits of the manufacturing number.

Project specific companion specifications may specify a different structure for the system title, but in any case, the system title shall be 8 octets long – if necessary, right padded to 8 octets – and shall be unique. The details shall be specified by the naming authority designated as such for the project.

Before the message security algorithms can be used – this requires a ciphered application context – the peers have to exchange system titles. The following possibilities are available:

- when the S-FSK PLC profile is used, system titles are exchanged during the registration process using the CIASE protocol. See IEC 62056-8-3;
- in all communication profiles, system titles are exchanged during AA establishment with a ciphered context using the COSEM-OPEN service – see 6.2 – carried the AARQ / AARE APDUs;

⁶ Administered by the DLMS UA acting as a registration authority; the same as the one used in the LDN.

If the system titles sent / received during AA establishment are not the same as the ones exchanged during the CIASE registration process, the AA shall be rejected.

- by writing the client_system_title attribute and by reading the server_system_title attribute of Security setup objects; see IEC 62056-6-2:—, 5.3.5.

NOTE 3 For broadcast communication, only the COSEM client sends it to the COSEM server.

5.4.8.3.4.7 Invocation field

The invocation field shall be an integer counter (frame counter).

5.4.8.3.5 Example

The following example (see Table 6) illustrates the process of creating ciphered xDLMS APDUs when applying authentication only, encryption only and authenticated encryption.

Table 6 – Example for ciphered APDUs

	<i>X</i>	Contents			<i>LEN</i> (<i>X</i>) bytes	<i>Len</i> (<i>X</i>) bits
Security material						
Security suite		GCM-AES-128				
System Title	<i>Sys-T</i>	4D4D4D0000BC614E(here, the five last octets contain the manufacturing number in hexadecimal form)			8	64
Frame Counter	<i>FC</i>	01234567			4	32
Initialization Vector	<i>IV</i>	<i>Sys-T</i> <i>FC</i>			12	96
		4D4D4D0000BC614E01234567				
Block cipher key (global)	<i>EK</i>	000102030405060708090A0B0C0D0E0F			16	128
Authentication Key	<i>AK</i>	D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF			16	128
Security applied		Authentication	Encryption	Authenticated encryption		
Security control byte (with unicast key)	<i>SC</i>	<i>SC-A</i>	<i>SC-E</i>	<i>SC-AE</i>	1	8
		10	20	30		
Security header	<i>SH</i>	<i>SH</i> = <i>SC-A</i> <i>FC</i>	<i>SH</i> = <i>SC-E</i> <i>FC</i>	<i>SH</i> = <i>SC-AE</i> <i>FC</i>		
		1001234567	2001234567	3001234567	5	40
Inputs		Authentication	Encryption	Authenticated encryption		
xDLMS APDU to be protected	<i>APDU</i>	C0010000080000010000FF0200 (Get-request, attribute 2 of the Clock object)			13	104
Plaintext	<i>P</i>	Null	C001000008000001 0000FF0200	C001000008000001 0000FF0200	13	104
Associated data	<i>A</i>	<i>SC</i> <i>AK</i> <i>APDU</i>	–	<i>SC</i> <i>AK</i>		
Associated Data – Authentication	<i>A-A</i>	10D0D1D2D3D4D5D6 D7D8D9DADBDCDDDE DFC0010000080000 010000FF0200	–	–	30	240
Associated Data – Encryption	<i>A-E</i>	–	–	–	0	0
Associated Data – Authenticated encryption	<i>A-AE</i>	–	–	30D0D1D2D3D4D5D6 D7D8D9DADBDCDDDE DF	17	136
Outputs		Authentication	Encryption	Authenticated encryption		
Ciphertext	<i>C</i>	NULL	411312FF935A4756 6827C467BC	411312FF935A4756 6827C467BC	13	104
Authentication tag	<i>T</i>	06725D910F9221D2 63877516	–	7D825C3BE4A77C3F CC056B6B	12	96
The complete Ciphered APDU		<i>TAG</i> <i>LEN</i> <i>SH</i> <i>APDU</i> <i>T</i>	<i>TAG</i> <i>LEN</i> <i>SH</i> <i>C</i>	<i>TAG</i> <i>LEN</i> <i>SH</i> <i>C</i> <i>T</i>	–	–
Authenticated APDU		C81E1001234567C0 0100000800000100 00FF020006725D91 0F9221D263877516	–	–	32	256
Encrypted APDU		–	C812200123456741 1312FF935A475668 27C467BC	–	20	160
Authenticated and encrypted APDU		–	–	C81E300123456741 1312FF935A475668 27C467BC7D825C3B E4A77C3FCC056B6B	32	256

5.4.8.4 High Level Security peer authentication with GMAC (authentication_mechanism_id(5))

NOTE See also 5.3.4.

In the case of authentication_mechanism_id(5), the process is the following:

- Pass 1 and Pass 2 shall be the same as described in 5.3.4;
- in Pass 3 and 4, instead of the HLS secret (held by the Association LN / Association SN objects), the global unicast encryption key, and the authentication key (if in use) are used. These are transferred to the server using the global_key_transfer method of the Security setup class. See IEC 62056-6-2:—, 5.3.5;
- in Pass 3, the client processes the challenge StoC. The authentication tag T is calculated using GMAC on $SC \parallel AK \parallel StoC$. The result $f(StoC) = SC \parallel FC \parallel T$ is sent back to the server. The server checks if $f(StoC)$ is the result of correct processing and – if correct – accepts the authentication of the client;
- in Pass 4: if the client is authenticated, the server processes CtoS in the same way as described in Pass 3. The authentication tag T is calculated using GMAC on $SC \parallel AK \parallel CtoS$. The result $f(CtoS) = SC \parallel FC \parallel T$ is sent back to the server. The client checks if $f(CtoS)$ is the result of correct processing and – if correct – accepts the authentication of the server.

An example is shown in Table 7 below.

Table 7 – HLS example with GMAC

Security material	<i>X</i>	Contents		<i>LEN(X)</i> bytes	<i>len(X)</i> bits
Security suite		GCM-AES-128			
System Title	<i>Sys-T</i>	Client	Server		
		4D4D4D0000000001	4D4D4D0000BC614E		
		(here, the five last octets contain the manufacturing number in hexadecimal form)		8	64
Frame Counter	<i>FC</i>	00000001	01234567	4	32
Initialization Vector	<i>IV</i>	<i>Sys-T</i> <i>FC</i>		12	96
		Client	Server		
		4D4D4D0000000001 00000001	4D4D4D0000BC614E 01234567		
Block cipher key (global)	<i>EK</i>	000102030405060708090A0B0C0D0E0F		16	128
Authentication Key	<i>AK</i>	D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF		16	128
Security control	<i>SC</i>	10		1	8
		Pass 1: Client sends challenge to server			
CtoS		4B35366956616759 "K56iVagY"		8	64
		Pass 2: Server sends challenge to client			
StoC		503677524A323146 "P6wRJ21F"		8	64
		Pass 3: Client processes StoC			
SC AK StoC		10D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF503677524A323146			
T (SC AK StoC)		1A52FE7DD3E72748973C1E28		12	96
f(StoC)		10000000011A52FE7DD3E72748973C1E28		17	136
		Pass 4: Server processes CtoS			
SC AK CtoS		10D0D1D2D3D4D5D6D7D8D9DADBDCDDDEDF4B35366956616759			
T (SC AK CtoS)		FE1466AFB3DBCD4F9389E2B7		12	96
f(CtoS)		1001234567FE1466AFB3DBCD4F9389E2B7		17	136

6 COSEM application layer service specification

6.1 Service primitives and parameters

In general, the services of a layer (or sublayer) are the capabilities it offers to a user in the next higher layer (or sublayer). In order to provide its service, a layer builds its functions on the services it requires from the next lower layer. Figure 7 illustrates this notion of service hierarchy and shows the relationship of the two correspondent N-users and their associated N-layer peer protocol entities.

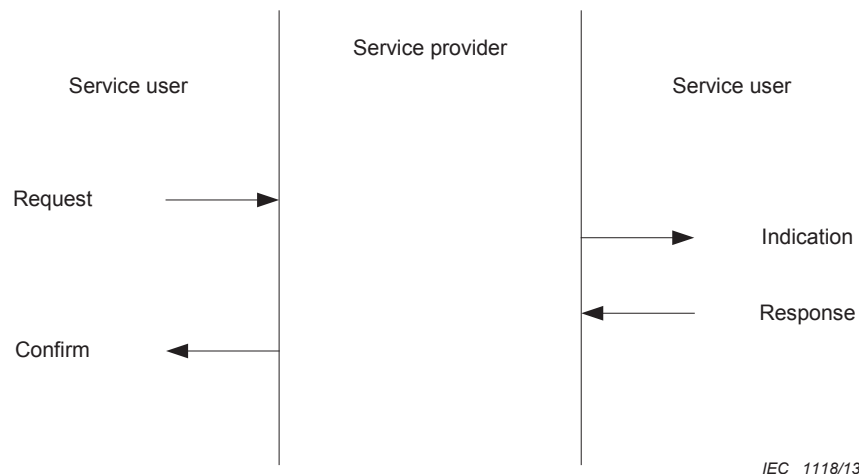


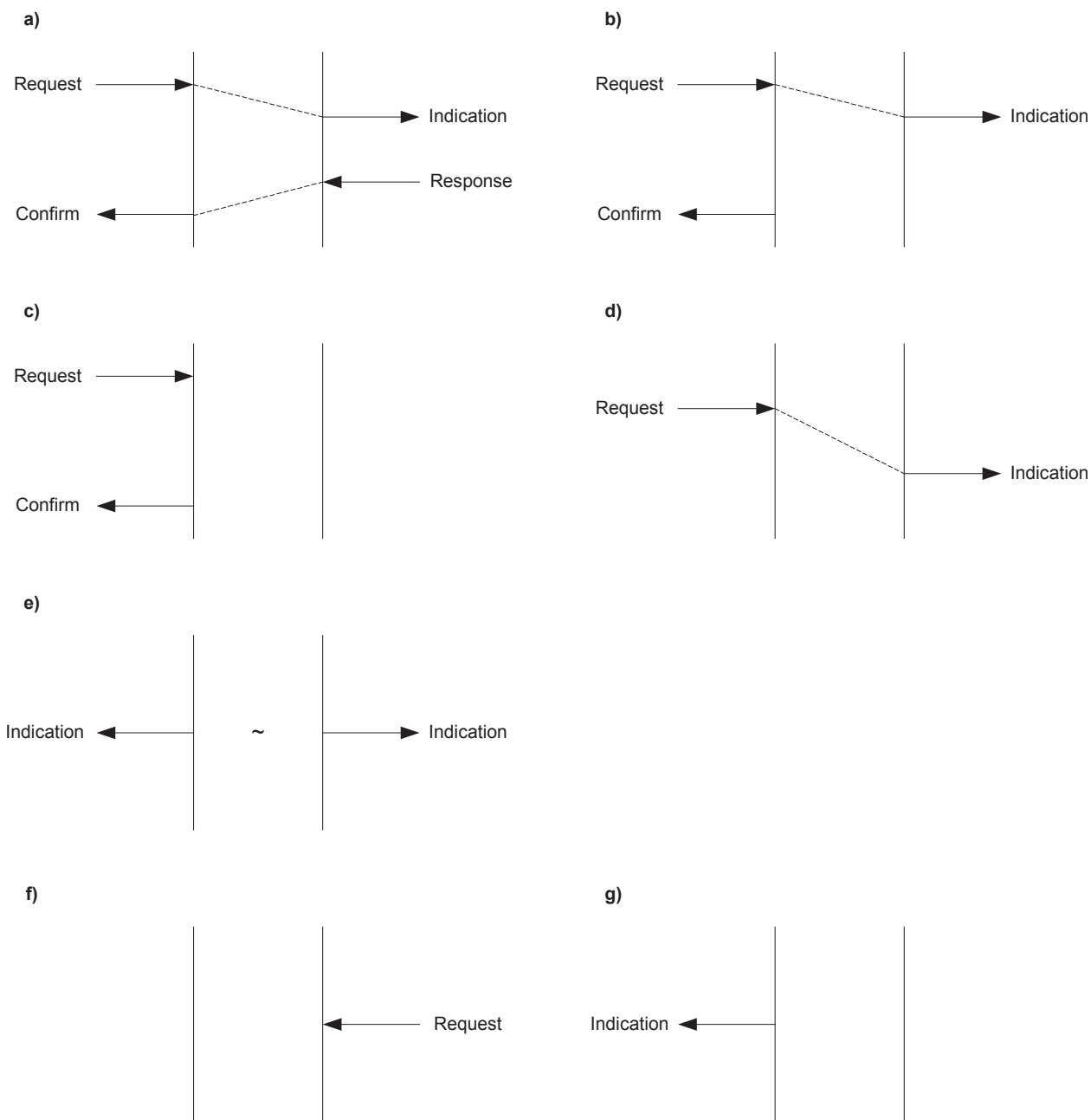
Figure 7 – Service primitives

Services are specified by describing the information flow between the N-user and the N-layer. This information flow is modelled by discrete, instantaneous events, which characterize the provision of a service. Each event consists of passing a service primitive from one layer to the other through an N-layer service access point associated with an N-user. Service primitives convey the information required in providing a particular service. These service primitives are an abstraction in that they specify only the service provided rather than the means by which the service is provided. This definition of service is independent of any particular interface implementation.

Services are specified by describing the service primitives and parameters that characterize each service. A service may have one or more related primitives that constitute the activity that is related to the particular service. Each service primitive may have zero or more parameters that convey the information required to provide the service. Primitives are of four generic types:

- **REQUEST:** The request primitive is passed from the N-user to the N-layer to request that a service be initiated;
- **INDICATION:** The indication primitive is passed from the N-layer to the N-user to indicate an internal N-layer event that is significant to the N-user. This event may be logically related to a remote service request, or may be caused by an event internal to the N-layer;
- **RESPONSE:** The response primitive is passed from the N-user to the N-layer to complete a procedure previously invoked by an indication primitive;
- **CONFIRM:** The confirm primitive is passed from the N-layer to the N-user to convey the results of one or more associated previous service request(s).

Possible relationships among primitive types are illustrated by the time-sequence diagrams shown in Figure 8. The figure also indicates the logical relationship of the primitive types. Primitive types that occur earlier in time and are connected by dotted lines in the diagrams are the logical antecedents of subsequent primitive types.



IEC 1119/13

Figure 8 – Time sequence diagrams

The service parameters of the COSEM AL service primitives are presented in a tabular format. Each table consists of two to five columns describing the service primitives and their parameters. In each table, one parameter – or a part of it – is listed on each line. In the appropriate service primitive columns, a code is used to specify the type of usage of the parameter. The codes used are listed in Table 8.

Some parameters may contain sub-parameters. These are indicated by labelling of the parameters as M, U, S or C, and indenting all sub-parameters under the parameter. Presence of the sub-parameters is always dependent on the presence of the parameter that they appear under. For example, an optional parameter may have sub-parameters; if the parameter is not supplied, then no sub-parameters may be supplied.

Table 8 – Codes for AL service parameters

M	The parameter is mandatory for the primitive.
U	The parameter is a user option, and may or may not be provided depending on dynamic usage by the ASE user.
S	The parameter is selected among other S-parameters as internal response of the server ASE environment.
C	The parameter is conditional upon other parameters or the environment of the ASE user.
(–)	The parameter is never present.
=	The " (=) " code following one of the M, U, S or C codes indicates that the parameter is semantically equivalent to the parameter in the service primitive to its immediate left in the table. For instance, an " M (=) " code in the .indication service primitive column and an "M" in the .request service primitive column means that the parameter in the .indication primitive is semantically equivalent to the one in the .request primitive.

Throughout this standard, the following rules are observed regarding the naming of terms:

- the name of ACSE services and the data transfer services using LN referencing is written in uppercase. Examples are: COSEM-OPEN, GET;
- the name of the data transfer services using SN referencing is written in title case. Examples are: Read, Write;
- camel notation is used in the following cases: EventNotification, TriggerEventNotificationSending, UnconfirmedWrite, InformationReport;
- the types of the LN service primitives may be mentioned in two alternative forms. Examples: "GET.request service primitive of Request_Type == NORMAL" or "GET-REQUEST-NORMAL service primitive";
- service parameter name elements are capitalized and joined with an underscore to signify a single entity: Examples are Protocol_Connection_Parameters and COSEM_Attribute_Descriptor;
- when the same parameter may occur several times, this is indicated by repeating the parameter in curly brackets. Example: Data { Data };
- in the data transfer service specifications, parameters used with block transfer only are shown in bold. Example: **DataBlock_G**;
- direct reference to a service parameter uses the capitalized form, while indirect (non-specific) reference uses the normal text without underscore joining. A direct reference example is: "The COSEM_Attribute_Descriptor parameter references a COSEM object attribute." An indirect (non-specific) reference example is: "A GET-REQUEST-NORMAL service primitive contains a single COSEM attribute descriptor";
- the names of COSEM data transfer APDUs using LN referencing are capitalized and joined with a dash to signify a single entity. Example: Get-Request-Normal;
- the names of COSEM data transfer APDUs using SN referencing use the camel notation. Example: ReadRequest.

6.2 The COSEM-OPEN service

NOTE The description has been amended to cover the new data transport security features. The order of the service parameters has been aligned with the order of the fields of the AARQ / AARE APDUs.

Function

The function of the COSEM-OPEN service is to establish an AA between peer COSEM APs. It uses the A-ASSOCIATE service of the ACSE. The COSEM-OPEN service provides only the framework for *transporting* this information. To provide and verify that information is the job of the appropriate COSEM AP.

Semantics of the service primitives

The COSEM-OPEN service primitives shall provide parameters as shown in Table 9.

Table 9 – Service parameters of the COSEM-OPEN service primitives

	.request	.indication	.response	.confirm
Protocol_Connection_Parameters	M	M (=)	M	M (=)
ACSE_Protocol_Version	U	U (=)	U	U (=)
Application_Context_Name	M	M (=)	M	M (=)
Called_AP_Title	U	U (=)	–	–
Called_AE_Qualifier	U	U(=)	–	–
Called_AP_Invocation_Identifier	U	U (=)	–	–
Called AE_Invocation_Identifier	U	U (=)	–	–
Calling_AP_Title	C	C (=)	–	–
Calling_AE_Qualifier	U	U (=)	–	–
Calling_AP_Invocation_Identifier	U	U (=)	–	–
Calling AE_Invocation_Identifier	U	U (=)	–	–
Local_Or_Remote	–	–	–	M
Result	–	–	M	M
Failure_Type	–	–	M	M
Responding_AP_Title	–	–	C	C (=)
Responding_AE_Qualifier	–	–	U	U (=)
Responding_AP_Invocation_Identifier	–	–	U	U (=)
Responding AE_Invocation_Identifier	–	–	U	U (=)
ACSE_Requirements	U	U (=)	U	U (=)
Security_Mechanism_Name	C	C (=)	C	C (=)
Calling_Authentication_Value	C	C (=)	–	–
Responding_Authentication_Value	–	–	C	C (=)
Implementation_Information	U	U (=)	U	U (=)
Proposed_xDLMS_Context	M	M (=)	–	–
Dedicated_Key	C	C (=)	–	–
Proposed_DLMS_Version_Number	M	M (=)	–	–
Proposed_DLMS_Conformance	M	M (=)	–	–
Client_Max_Receive_PDU_Size	M	M (=)	–	–
Negotiated_xDLMS_Context	–	–	S	S (=)
Negotiated_DLMS_Version_Number	–	–	M	M (=)
Negotiated_DLMS_Conformance	–	–	M	M (=)
Server_Max_Receive_PDU_Size	–	–	M	M (=)
VAA_Name			M	M (=)
XDLMs_Initiate_Error			S	S (=)
User_Information	U	C (=)	–	–
Service_Class	M	M (=)	–	–

The service parameters of the COSEM-OPEN.request service primitive, except the Protocol_Connection_Parameters, the User_Information parameter and – depending on the communication profile – the Service_Class parameter are carried by the fields of the AARQ APDU sent by the client.

The service parameters of the COSEM-OPEN.response service primitive, except the Protocol_Connection_Parameters is carried by the fields of the AARE APDU sent by the server.

The A-ASSOCIATE service and the AARQ and AARE APDUs are specified in 7.2. Encoding examples are given in Annex B.

The Protocol_Connection_Parameters parameter is mandatory. It contains all information necessary to use the layers of the communication profile, including the communication profile (protocol) identifier and the addresses required. It identifies the participants of the AA. The elements of this parameter are passed to the entities managing lower layer connections and to the lower layers as appropriate.

The ACSE_Protocol_Version parameter is optional. If present, the default value shall be used.

The Application_Context_Name parameter is mandatory. In the request primitive, it holds the value proposed by the client. In the response primitive, it holds the same value or the value supported by the server.

The Calling_AP_Title parameter is conditional. When the Application_Context_Name indicates an application context using ciphering, it shall carry the client system title specified in 5.4.8.3.4.6.

The use of the Called_AP_Title, Called_AE_Qualifier, Called_AP_Invocation_Identifier, Called_AE_Invocation_Identifier, Calling_AE_Qualifier, Calling_AP_Invocation_Identifier and Calling_AE_Invocation_Identifier parameters is optional. Their use is not specified in this companion specification.

The Local_or_Remote parameter is mandatory. It indicates the origin of the COSEM-OPEN.confirm primitive. It is set to Remote if the primitive has been generated following the reception of an AARE APDU from the server. It is set to Local if the primitive has been locally generated.

The Result parameter is mandatory. In the case of remote confirmation, it indicates whether the Server accepted the proposed AA or not. In the case of local confirmation, it indicates whether the client side protocol stack accepted the request or not.

The Failure_type parameter is mandatory. In the case of remote confirmation, it carries the information provided by the server. In the case of local and negative confirmation, it indicates the reason for the failure.

The use of the Responding_AP_Title parameter is conditional. When the Application_Context_Name parameter indicates an application context using ciphering, it shall carry the server system title specified in 5.4.8.3.4.6.

The use of the Responding_AE_Qualifier, Responding_AP_Invocation_Identifier and Responding_AE_Invocation_Identifier parameters is optional. Their use is not specified in this companion specification.

The ACSE_Requirements parameter is optional. It is used to select the optional authentication functional unit of the A-Associate service for the association; see 7.2.1.

The presence of the ACSE_Requirements parameter depends on the security mechanism used:

- in the case of Lowest Level security, it shall not be present; only the Kernel functional unit will be used;

- in the case of Low Level security (LLS) it shall be present in the .request primitive and it may be present in the .response service primitive;
- in the case of High Level Security (HLS), it shall be present both in the .request and the .response service primitives.

The `Security_Mechanism_Name` parameter is conditional. It is present only if the authentication functional unit has been selected. If present, the .request primitive holds the value proposed by the client and the .response primitive holds the value required by the server, i.e. the one to be used by the client.

The `Calling_Authentication_Value` parameter and the `Responding_Authentication_Value` parameters are conditional. They are present only if the authentication functional unit has been selected. They hold the client authentication value / server authentication value respectively, appropriate for the `Security_Mechanism_Name`.

The `Implementation_Information` parameter is optional. Its use is not specified in this companion specification.

The `Proposed_xDLMS_Context` parameter holds the elements of the proposed xDLMS context carried by the xDLMS InitiateRequest APDU, placed in the user-information field of the AARQ APDU.

The `Dedicated_Key` element is conditional. It may be present only, when the `Application_Context_Name` parameter indicates an application context using ciphering. The dedicated key is used for dedicated ciphering of xDLMS APDUs exchanged within the AA established.

When the dedicated key is present, the xDLMS InitiateRequest APDU shall be authenticated and encrypted using the AES-GCM-128 algorithm, the global unicast encryption key and the authentication key (if in use). The xDLMS InitiateRequest APDU shall be ciphered the same way, when the dedicated key is not present, but it is necessary to protect the RLRQ APDU by including the ciphered xDLMS InitiateRequest in its user-information field; see 6.3.

The `Proposed_DLMS_Version_Number` element holds the proposed DLMS version number; see 4.2.3.1.

The `Proposed_DLMS_Conformance` element holds the proposed conformance block; see 7.3.1.

The `Client_Max_Receive_PDU_Size` element holds the maximum length of the xDLMS APDUs the client can receive; see Table 1.

If the xDLMS context proposed by the client is acceptable for the server, then the response service primitive shall contain the `Negotiated_xDLMS_Context` parameter. It holds the elements of the negotiated xDLMS context, carried by the xDLMS InitiateResponse APDU, placed in the user-information field of the AARE APDU. If the xDLMS InitiateRequest APDU has been ciphered, the xDLMS InitiateResponse APDU shall be also ciphered the same way.

The `Negotiated_DLMS_Version_Number` element holds the negotiated DLMS version number. See 4.2.3.1.

The `Negotiated_DLMS_Conformance` element holds the negotiated conformance block. See 7.3.1.

The `Server_Max_Receive_PDU_Size` element carries the maximum length of the xDLMS APDUs the server can receive; see Table 1.

The VAA_name element carries the dummy value of 0x0007 in the case of LN referencing, and the base_name of the current Association object, 0xFA00, in the case of SN referencing.

If the xDLMS context proposed by the client is not acceptable for the server, then the response service primitive shall carry the xDLMS_Initiate_Error parameter. It is carried by the ConfirmedServiceError APDU, with appropriate diagnostic elements, placed in the user-information field of the AARE APDU.

The User_Information parameter is optional. If present, it shall be passed on to the supporting layer, provided it is capable to carry it. The indication primitive shall then contain the user-specific information carried by the supporting lower protocol layer(s); see Annex A.

The Service_Class parameter is mandatory. It indicates whether the service shall be invoked in a confirmed or in an unconfirmed manner. The handling of this parameter may depend on the communication profile; see Annex A.

Use

A possible logical sequence of the COSEM-OPEN service primitives is illustrated in Figure 8:

- for confirmed AA – successful or unsuccessful – establishment, item a);
- for unconfirmed AA establishment, item b);
- in the case of a pre-established AA or an unsuccessful attempt due to a local error, item c).

The .request primitive is invoked by the client AP to request the establishment of a confirmed or an unconfirmed AA with a server AP ⁷. Upon the reception of the request invocation, the AL constructs and sends an AARQ APDU to the server.

The .indication primitive is generated by the server AL when a correctly formatted AARQ APDU is received.

The .response primitive is invoked by the server AP to indicate to the AL whether the proposed AA is accepted or not. It is invoked only if the proposed AA is confirmed. The AL constructs then an AARE APDU and sends it to its peer, containing the service parameters received from the AP.

The .confirm primitive is generated by the client AL to indicate to the client AP whether the AA previously requested is accepted or not:

- remotely, when an AARE APDU is received;
- locally, if the requested AA already exists; this includes pre-established AAs;
- locally, if the corresponding .request primitive has been invoked with Service_Class == Unconfirmed;
- locally, if the requested AA is not allowed;
- locally, if an error is detected: missing or not correct parameters, failure during the establishment of the requested lower layer connections, missing physical connection, etc.

The protocol for establishing an AA is specified in 7.2.4. Communication profile specific rules are specified in Annex A.

⁷ Before the invocation of the COSEM-OPEN.request primitive, the physical layers shall be connected. Depending on the communication profile, the invocation of this primitive may also imply the connection of other lower layers.

6.3 The COSEM-RELEASE service

NOTE 1 The description has been amended to better align with the A-Release service and to allow the user-information field of the RLRQ / RLRE to carry a ciphered xDLMS InitiateRequest / InitiateResponse APDU, to secure the COSEM-RELEASE service.

Function

The function of the COSEM-RELEASE service is to gracefully release an existing AA. Depending on the way it is invoked, it uses the A-RELEASE service of the ACSE or not.

Semantics of the service primitives

The COSEM-RELEASE service primitives shall provide parameters as shown in Table 10.

Table 10 – Service parameters of the COSEM-RELEASE service primitives

	.request	.indication	.response	.confirm
Use_RLRQ_RLRE	U	C(=)	C(=)	-
Reason	U	U (=)	U	U (=)
Proposed_xDLMS_Context	C	C (=)	–	–
Negotiated_xDLMS_Context	–	–	C	C (=)
Local_Or_Remote	–	–	–	M
Result	–	–	M	M
Failure_Type	–	–	–	C
User_Information	U	C (=)	U	C (=)

The Use_RLRQ_RLRE parameter in the .request primitive is optional. If present, its value may be FALSE (default) or TRUE. It indicates whether the ACSE A-RELEASE service – involving an RLRQ / RLRE APDU exchange – should be used or not. The A-RELEASE service and the RLRQ / RLRE APDUs are specified in 7.2. The Use_RLRQ_RLRE parameter in the .response primitive is conditional. If it was present in the .indication primitive and if its value was TRUE, it shall also be present and its value shall be TRUE. Otherwise, it shall not be present or its value shall be FALSE.

If the value of the Use_RLRQ_RLRE parameter is FALSE, then the AA can be released by disconnecting the supporting layer of the AL.

The Reason parameter is optional. It may be present only if the value of the Use_RLRQ_RLRE is TRUE. It is carried by the reason field of the RLRQ / RLRE APDU respectively.

When used on the .request primitive, this parameter identifies the general level of urgency of the request. It takes one of the following symbolic values:

- normal;
- urgent (not available in DLMS/COSEM); or
- user defined.

When used on the .response primitive, this parameter identifies information about why the acceptor accepted or rejected the release request. Note that in DLMS/COSEM, the server cannot reject the release request. It takes one of the following symbolic values:

- normal;
- not finished; or