ETSI GS ECI 001-5-2 V1.1.1 (2017-07)

Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 5: The Advanced Security System; Sub-part 2: Key Ladder Block

Disclaimer

The present document has been produced and approved by the Embedded Common Interface (ECI) for exchangeable CA/DRM solutions ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference
DGS/ECI-001-5-2

Keywords
CA, DRM, swapping

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

List of Figures

List of Tables

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Group Specification (GS) has been produced by ETSI Industry Specification Group (ISG) Embedded Common Interface (ECI) for exchangeable CA/DRM solutions.

The present document is part 5, sub-part 2 of a multi-part deliverable covering the ECI specific functionalities of an advanced security system, as identified below:

Part 1: "Architecture, Definitions and Overview";

Part 2: "Use cases and requirements";

Part 3: "CA/DRM Container, Loader, Interfaces, Revocation";

Part 4: "The Virtual Machine";

Part 5: "The Advanced Security System:

Sub-part 1: "ECI specific functionalities";

Sub-part 2: "Key Ladder Block".

Part 6: "Trust Environment".

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

Introduction

A **content provider** encrypts their digital content and uses a **content protection system** in order to protect the content against unauthorized access. A consumer uses a **content receiver** to access protected content. To this end, the **content receiver** contains a chipset that implements one or more content decryption operations. A cryptographic key establishment protocol is used to secure the transport of content decryption keys from the **content protection system** to the chipset. The steps of the protocol that are implemented within the chipset are referred to as a key ladder in the present document. The present document specifies a key ladder for the key establishment protocol presented in [i.1].

The key ladder and the protocol may also be used to secure the transport of content encryption keys to the chipset. Such keys are required for use cases in which the chipset re-encrypts content. The chipset may implement one or more content encryption operations for this purpose. Personal video recording and exporting protected content to a different **content protection system** are typical examples of content re-encryption use cases. Content decryption keys and content encryption keys are both referred to as **control word**s throughout the present document.

The present document also specifies an authentication mechanism. This mechanism is closely related to the key ladder and may be used for entity authentication; in other words, this mechanism may be used to authenticate the chipset.

The key ladder and authentication mechanism specified in the present document are agnostic to both the **content protection system** and the **content provider**. This enables a **content provider** to use any compliant **content protection system**, and it enables a consumer to use the **content receiver** for accessing content of any **content provider** that uses a compliant **content protection system**.

A **certification authority** manages a public-key certificate of each chipset in the mechanisms specified in the present document. In particular, the **certification authority** distributes such certificates and certificate revocation information to **content provider**s that want to make use of the key ladder and/or the authentication mechanism. Next, the **content provider**s use the certificates and certificate revocation information as input to their compliant **content protection system**; as detailed later, the knowledge of the public key in the certificate of a chipset enables the **content protection system** to generate suitable input messages for the chipset's key ladder and authentication mechanism.

1 Scope

The present document specifies a key ladder block for implementation in a **content receiver**'s chipset. The key ladder block comprises a key ladder for securing the transport of **control word**s to the chipset and an authentication mechanism. The present document also specifies aspects of the personalization of a compliant chipset.

The present document is intended for use by chipset manufacturers.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at https://docbox.etsi.org/Reference.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] IEEE Standards Association™: "Guidelines for Use Organizationally Unique Identifier (OUI) and Company ID (CID)".

NOTE: Available at https://standards.ieee.org/develop/regauth/tut/eui.pdf.

[2] RSA Laboratories: "PKCS #1 v2.2: RSA Cryptography Standard".

[3] NIST FIPS PUB 197: "Specification for the Advanced Encryption Standard (AES)".

[4] NIST FIPS PUB 180-4: "Secure Hash Standard (SHS)".

[5] NIST SP 800-107 Revision 1: "Recommendation for Applications Using Approved Hash Algorithms".

[6] ISO/IEC 9797-1:2011: "Information technology -- Security techniques -- Message Authentication Codes (MACs) -- Part 1: Mechanisms using a block cipher".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] P. Roelse: "A new key establishment protocol and its application in pay-TV systems".

[i.2] ETSI TS 100 289: "Digital Video Broadcasting (DVB); Support for use of the DVB Scrambling Algorithm version 3 within digital broadcasting systems".

[i.3] ETSI TS 103 127: "Digital Video Broadcasting (DVB); Content Scrambling Algorithms for DVB-IPTV Services using MPEG2 Transport Streams".

[i.4] ATSC Standard A/70 Part 1:2010: "Conditional Access System for Terrestrial Broadcast".

[i.5] ISO/IEC 23001-7:2016: "Information technology -- MPEG systems technologies -- Part 7: Common encryption in ISO base media file format files".

[i.6] Radio, Film and Television Industrial Standard of the People's Republic of China GY/T 277 - 2014: "Technical Specification of Digital Rights Management for Internet Television".

NOTE: This reference is only available in Chinese.

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

certification authority: party that is responsible for managing public-key certificates

NOTE: A **certification authority** is trusted by all other parties in the system to perform operations associated with certificates.

chipset-ID: non-secret number that is used to identify a chipset

content protection system: system that uses cryptographic techniques to manage access to digital content

NOTE: Typically, a **content protection system** is either a conditional access system or a digital rights management system.

content provider: party that distributes digital content to a **content receiver**

content receiver: device that is used to access digital content

NOTE: A **content receiver** contains a chipset with a **content descrambler**.

content descrambler: component in the chipset that is capable of decrypting content

NOTE: A **content descrambler** may also be capable of encrypting content (for the purpose of content re-encryption). In the present document, content encryption/decryption uses a **symmetric encryption scheme**. For MPEG-2 content, content encryption and decryption are also referred to as scrambling and descrambling, respectively.

control word: secret key used to encrypt and decrypt content

NOTE: In digital rights management systems, a **control word** is typically referred to as a content key.

cryptographic hash function: unkeyed cryptographic function that takes data of arbitrary size, referred to as the message, as input and produces an output data block of fixed size, referred to as the message digest

NOTE: Assumed properties of the **cryptographic hash function** in the present document are:

1) the **cryptographic hash function** behaves as a random function; and

2) the **cryptographic hash function** is second preimage resistant.

digital signature scheme: keyed asymmetric cryptographic scheme that is used to protect the authenticity of data

NOTE: A **digital signature scheme** consists of a key generation algorithm, a signature generation operation and a signature verification operation. Keys are generated as (secret/private key, public key) pairs. The data is signed using a secret/private key and the corresponding public key is used to verify the signature. The **digital signature scheme** specified in the present document is used to protect the authenticity of messages as defined in [i.1]; in particular, the scheme is not used to provide non-repudiation or source authentication in the present document.

message authentication code algorithm: keyed symmetric cryptographic algorithm that is used to protect the authenticity of data

NOTE: A **message authentication code algorithm** takes a message and a secret key as inputs, and produces an output data block referred to as the MAC. The **message authentication code algorithm** as specified in the present document is used to cryptographically bind a ciphertext message to its associated data; in particular, the algorithm is not used to provide source authentication in the present document.

public-key encryption scheme: keyed asymmetric cryptographic scheme that is used to protect the confidentiality of data

NOTE: A **public-key encryption scheme** consists of a key generation algorithm, an encryption operation and a decryption operation. Keys are generated as (public key, secret/private key) pairs. Data is encrypted using a public key and the data is recovered from the ciphertext using the corresponding secret/private key.

symmetric encryption scheme: keyed symmetric cryptographic scheme that is used to protect the confidentiality of data

NOTE: A **symmetric encryption scheme** consists of a key generation algorithm, an encryption operation and a decryption operation. The encryption and decryption operations of a **symmetric encryption scheme** use the same secret key as input.

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AES	Advanced Encryption Standard
AD1	Associated Data 1
AD2	Associated Data 2
AK	Authentication Key
ATSC	Advanced Television Systems Committee
CA/DRM	Conditional Access/Digital Rights Management
CID	Company IDentifier
CISSA	Common IPTV Software-oriented Scrambling Algorithm
CPU	Central Processing Unit
CSA	Common Scrambling Algorithm
CPK	Chipset Public Key
CSK	Chipset Secret/private Key
CW	Control Word
DVB	Digital Video Broadcasting
ECB	Electronic Code Book
ID	IDentity
Len	Length
LK	Link Key
MAC	Message Authentication Code
MK	MAC Key
MPEG	Moving Pictures Expert Group
OUI	Organizationally Unique Identifier
RSA	Rivest Shamir Adleman
SHA	Secure Hash Algorithm
SIM	Signed Input Message
SPK	Sender Public Key
SSK	Sender Secret/private Key
T	Tag
URI	Usage Rules Information

4 Chipset-ID and chipset master key pair

This clause specifies aspects of the personalization of a compliant chipset. Each compliant chipset is associated with a bit string that is used to identify the chipset, referred to as **chipset-ID**, and a chipset master key pair.

A globally unique 64-bit **chipset-ID** shall be allocated to every compliant chipset. If the bits of **chipset-ID** are numbered 0 to 63 from left to right and if the i^{th} bit ($0 \leq i \leq 63$) is denoted by b_i, then (b_0, b_1, b_2, b_3) shall contain a registration authority identifier. Each value of the registration authority identifier shall be associated with at most one registration authority. The **chipset-ID** shall also contain a chipset manufacturer identifier. The value of the registration authority identifier and the chipset manufacturer identifier of a compliant chipset's **chipset-ID** shall uniquely identify the chipset manufacturer that produced the chipset. In addition, the registration authority identified by the value of (b_0, b_1, b_2, b_3) shall administer the assignment of chipset manufacturer identifiers that can be used in combination with this value.

If (b_0, b_1, b_2, b_3) = (0, 0, 0, 0), then the IEEE Registration Authority (https://standards.ieee.org/develop/regauth/) shall be the registration authority and the 24-bit OUI (http://standards.ieee.org/develop/regauth/oui/) or the 24-bit CID (http://standards.ieee.org/develop/regauth/cid/) shall be used to identify chipset manufacturers. In addition, if (b_0, b_1, b_2, b_3) = (0, 0, 0, 0), then (b_4, b_5, …, b_{27}) shall contain the OUI/CID, b_4 being the most significant bit of Octet 0 of the OUI/CID (see also [1]) and b_{27} being the least significant bit of Octet 2 of the OUI/CID.

All other values of the registration authority identifier are reserved for future use.

The chipset master key pair is associated with a public-key encryption scheme, and consists of a chipset secret/private key CSK and a chipset public key CPK. As detailed later in clauses 5 and 6, (CSK, CPK) is the master key pair of both the key ladder and the authentication mechanism. The **public-key encryption scheme** and the representations of CSK and CPK are specified in clause 8.2.

A compliant chipset should generate its own key pair (CSK, CPK) to prevent that the value of CSK is known to any party in the system. In this case only **chipset-ID** needs to be distributed to the chipset during its personalization, and the authenticity of **chipset-ID** shall be protected during this distribution. If the chipset does not generate its own key pair, then the authenticity of the triple (**chipset-ID**, CSK, CPK) and the confidentiality of CSK shall be protected during the distribution of (**chipset-ID**, CSK, CPK) to the chipset.

The random number used as input to the (CSK, CPK) key pair generation algorithm shall have at least 128 bits of entropy.

A compliant chipset shall permanently store its triple (**chipset-ID**, CSK, CPK), and a compliant chipset shall implement measures to protect the confidentiality of the stored CSK and the integrity of the stored **chipset-ID**, CSK and CPK.

The CPU of the **content receiver** shall have read access to the stored **chipset-ID** and the stored chipset public key CPK. This enables a **certification authority** to use the exported information as input to create a public-key certificate for the chipset. In addition, the exported **chipset-ID** enables a **content provider** to identify the chipset and its certificate.

A **certification authority** shall maintain the pair (**chipset-ID**, CPK) for every chipset it manages. The present document does not exclude the presence of more than one **certification authority** in the system. The authenticity of the pair (**chipset-ID**, CPK) shall be protected during its distribution to the associated **certification authority** or authorities.

5 Key ladder

5.1 Overview

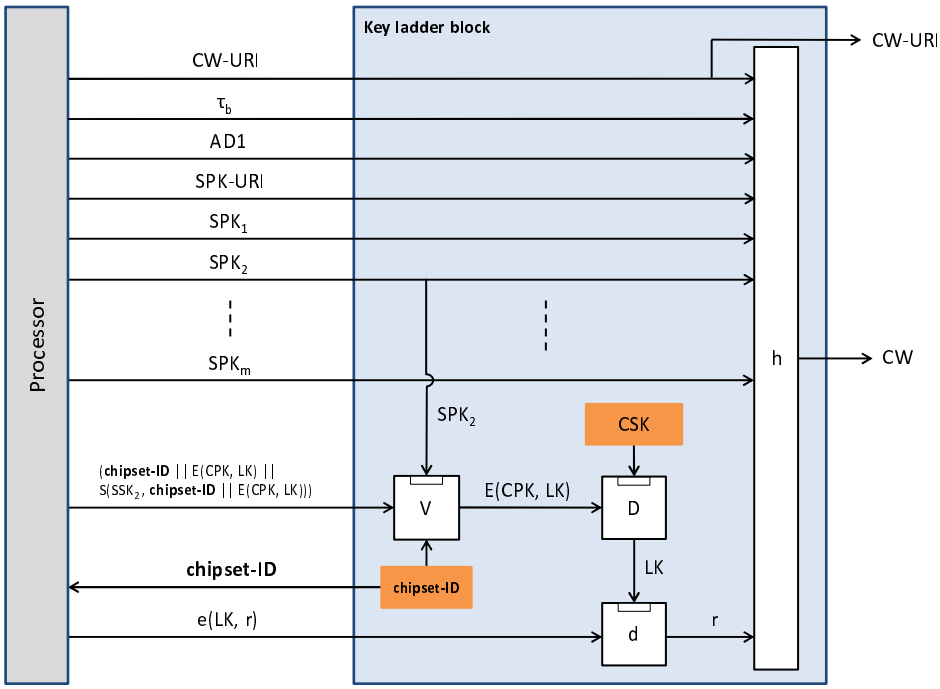

Figure 5.1-1: Key ladder

This clause presents the functional design of the key ladder. The block in the chipset that implements the key ladder is referred to as the key ladder block throughout the present document. The key ladder is depicted in Figure 5.1-1. As specified in clause 4 and as shown in the figure, the chipset is personalized with a **chipset-ID** and with a chipset secret/private key CSK.

One of the outputs of the key ladder block is a **control word** denoted by CW. CW is used for either content decryption or content encryption. A second output of the key ladder block is a bit string denoted by CW-URI. CW-URI defines usage rules information for CW (refer to clause 5.3.1 for the specification of CW-URI and the associated usage rule). CW and CW-URI are inputs to the **content descrambler** (the **content descrambler** is not depicted in Figure 5.1-1).

- The key ladder block and the **content descrambler** shall be implemented in a single silicon chip.
- If the **content descrambler** offers an interface to a processor in the **content receiver** that allows the processor to pass plaintext **control word**s to the **content descrambler** (i.e. by-passing the key ladder block), then it shall be possible to permanently disable this functionality.
- If the key ladder computes a CW, then only the key ladder block and the **content descrambler** shall have access to this CW.
- The authenticity of the pair (CW, CW-URI) and the confidentiality of CW shall be protected during their distribution from the key ladder block to the **content descrambler**.

The key ladder block shall interface with a processor of the **content receiver**. For example, the processor may be a security processor or the CPU of the **content receiver**. As specified in clause 4 and as shown in Figure 5.1-1, this processor has read access to the **chipset-ID**. This enables a **content provider** to identify the chipset and obtain the corresponding public-key certificate containing CPK from the **certification authority**. The value of CPK needs to be known to compute one of the input messages to the key ladder, as detailed in clause 5.2.

As specified in clause 4, the pair (CSK, CPK) is associated with a **public-key encryption scheme**. The corresponding encryption and decryption operations are denoted by E and D, respectively. E and D are keyed cryptographic operations and each of these operations has two inputs: a key input and a message input. The present document assumes that the first input of a keyed cryptographic operation or algorithm is the key.

EXAMPLE: The encryption of a message M using E and chipset public key CPK is written as E(CPK, M).

The mechanisms specified in the present document also use a **digital signature scheme**; S and V denote the signature generation operation and the signature verification operation, respectively. A key pair of the **digital signature scheme** is associated with a sender and consists of a sender secret/private key SSK and a sender public key SPK. The present document assumes that a sender is a **content protection system**. As shown in Figure 5.1-1, a number of different SPKs, denoted by SPK_1, SPK_2, ..., SPK_m ($m \geq 1$) are input to the key ladder block. The key ladder block shall support key ladders for all values of m with $1 \leq m \leq 16$. In practice, each key pair (SSK_i, SPK_i) with $1 \leq i \leq m$ will typically be associated with one **content protection system**; however, such a key pair may also be shared between multiple systems.

Associated with SPK_1, SPK_2, ..., SPK_m is SPK-URI. This input to the key ladder block defines the usage rules information for SPK_1, SPK_2, ..., SPK_m. SPK-URI and the associated usage rules are specified in clause 5.3.2. As shown in Figure 5.1-1, one of the SPKs and the verification operation V are used to verify the signature of the input message (**chipset-ID** || E(CPK, LK) || S(SSK_i, **chipset-ID** || E(CPK, LK))). This input message is also denoted by SIM_{KL} in the following text. Figure 5.1-1 assumes that i = 2 and that SPK-URI and the usage rules allow SPK_2 to be used for the verification of the signature.

The key ladder also implements a **symmetric encryption scheme**. The encryption and decryption operations of this scheme are denoted by e and d, respectively. The key ladder uses a link key LK as the key of this scheme and a random number r (or another link key as detailed later in clause 5.4) as the message. The random number r is represented as a bit string and its length shall be 128 bits.

Finally, the key ladder block implements a function h. This function is based on a **cryptographic hash function** (see clause 8.4 for the specification of h).

As shown in Figure 5.1-1, the other two inputs to the key ladder block are τ_b and Associated Data 1; the latter is also referred to as AD1. Both these inputs are represented as bit strings:

- The length of τ_b shall be 8 bits. The use of τ_b is specified in clause 5.5.
- The length of AD1 shall be 256 bits. The specification of the contents of AD1 is outside the scope of the present document and the present document assumes that the key ladder block does not process AD1 other than providing it as input to h. The key ladder block may pass AD1, or part of AD1, together with CW-URI and CW to the **content descrambler**.

The present document assumes that symmetric keys, ciphertext messages, and signatures are represented as bit strings. Their lengths are defined in clause 8. The representation of asymmetric keys is also specified in that clause.

5.2 Key ladder computations

The sender associated with key pair (SSK_i, SPK_i) for some $1 \leq i \leq m$ can create the signed input message SIM_{KL}, i.e.:

(**chipset-ID** || E(CPK, LK) || S(SSK_i, **chipset-ID** || E(CPK, LK)))

using the following steps.

Compute SIM_{KL} (sender):

1) Generate a link key LK.
2) Compute the ciphertext E(CPK, LK).
3) Concatenate **chipset-ID** and E(CPK, LK); the resulting bit string is denoted by (**chipset-ID** || E(CPK, LK)).
4) Sign the bit string (**chipset-ID** || E(CPK, LK)) using SSK_i; the signature is denoted by S(SSK_i, **chipset-ID** || E(CPK, LK))).
5) Append this signature to the bit string (**chipset-ID** || E(CPK, LK)).

After receiving SIM_{KL} and sender public key SPK_i, the key ladder block shall perform the following steps to retrieve LK (in Figure 5.1-1, it is assumed that the first three steps are performed by V).

Compute LK (key ladder block):

1) Verify if the received **chipset-ID** equals the stored **chipset-ID**. If these two values are not equal, then the key ladder block shall abort the computations.

2) Check if SPK-URI and the usage rules as defined in clause 5.3.2 allow V to use SPK_i to verify the signature. If this is not allowed, then the key ladder block shall abort the computations.

3) Use the received SIM_{KL} and SPK_i to verify the signature. If the signature is invalid, then the key ladder block shall abort the computations.

4) Compute LK = D(CSK, E(CPK, LK)).

Next, the key ladder block shall use LK to process the input message e(LK, r) (see also Figure 5.1-1). The sender can create this message using the following steps.

Compute e(LK, r) (sender):

1) Generate a random bit string r.

2) Compute e(LK, r).

After receiving e(LK, r) and after computing LK, the key ladder block shall compute r using the following step (see also Figure 5.1-1).

Compute r (key ladder block):

1) r = d(LK, e(LK, r)).

Next, the key ladder block shall use the function h to compute the **control word** CW. As shown in Figure 5.1-1, the inputs to h are CW-URI, τ_b, AD1, SPK-URI, SPK_1, SPK_2, ..., SPK_m, and r. The implementation of the key ladder shall ensure that the public key that was used to verify the authenticity of SIM_{KL} (or more precisely, the SIM_{KL} containing the link key associated with the random number r) is provided as one of the SPK-inputs to h when CW is derived from r. Next, the key ladder block shall pass CW-URI and CW to the **content descrambler**.

5.3 Usage Rules Information

5.3.1 CW-URI

The length of CW-URI shall be 64 bits and these bits are numbered 0 to 63 from left to right. The value of CW-URI defines the allowed usage of CW (see also Table 5.3.1-1) according to the following usage rule: if the value of a bit is 1, then the specified use is allowed, otherwise this use is not allowed. The **content descrambler** shall enforce this usage rule. The **content descrambler** shall ignore:

i) the value of the bits that are reserved for future use; and

ii) the value of bits that correspond to implementations that the **content descrambler** does not support.

Table 5.3.1-1: Definition of CW-URI

Bit number	Description
0	Encrypt
1	Decrypt
2 … 7	Reserved for future use
8	DVB CSA2 [i.2]
9	DVB CSA3 [i.2]
10 … 15	Reserved for future use
16	DVB CISSA version 1 [i.3]
17 … 23	Reserved for future use
24	ATSC common scrambling/descrambling algorithm [i.4]
25 … 31	Reserved for future use
32	Common encryption scheme - cenc protection scheme [i.5]
33	Common encryption scheme - cbc1 protection scheme [i.5]
34	Common encryption scheme - cens protection scheme [i.5]
35	Common encryption scheme - cbcs protection scheme [i.5]
36 … 39	Reserved for future use
40	ChinaDRM - cipher block chaining mode [i.6]
41	ChinaDRM - counter mode [i.6]
42 … 63	Reserved for future use

5.3.2 SPK-URI

The length of SPK-URI shall be 64 bits and these bits are also numbered 0 to 63 from left to right. SPK-URI is defined in Table 5.3.2-1. In particular, if SPK_1, SPK_2, ..., and SPK_m are provided as inputs to the key ladder together with SPK-URI (as depicted in Table 5.3.2-1), then Bits 0, 1, …, m-1 and Bits 16, 17, .., 16+m-1 of SPK-URI are used to define two subsets of {SPK_1, SPK_2, ..., SPK_m}: SPK_i (i = 1, 2, …, m) is an element of the first subset if and only if the value of Bit i-1 equals one, and SPK_i is an element of the second subset if and only if the value of Bit i+15 equals one. In the following text, these two subsets are denoted by S_1 and S_2, respectively. Using this notation, the key ladder shall apply the following usage rule:

SPK usage rule 1: in the key ladder, V is allowed to use SPK_i to verify the signature of SIM_{KL} if and only if $SPK_i \in S_1$.

SPK usage rule 1 shall be enforced by the key ladder block.

The set S_2 is used in the case of content re-encryption. If content is re-encrypted, then two sets of inputs as depicted in Figure 5.1-1 are required: one set of inputs associated with the CW used for content decryption and one set of inputs associated with the CW used for content encryption. In this case, the following usage rule connects these two sets of inputs:

SPK usage rule 2: in the key ladder, V is allowed to use SPK_i to verify the signature of SIM_{KL} of the CW used for content encryption if and only if $SPK_i \in S_2$ associated with the CW used for content decryption.

If the **content descrambler** supports content re-encryption, then the key ladder block should enforce SPK usage rule 2. If the key ladder block does not enforce this usage rule, then another component in the chipset shall enforce SPK usage rule 2 and the implementation shall ensure that the value of SPK-URI that is input to the function h is equal to the value of SPK-URI that is used to enforce SPK usage rule 2.

Table 5.3.2-1: Definition of SPK-URI

Bit number	Description
0	spk1_in_set1
1	spk2_in_set1
2	spk3_in_set1
…	…
15	spk16_in_set1
16	spk1_in_set2
17	spk2_in_set2
18	spk3_in_set2
…	…
31	spk16_in_set2
32 … 63	Reserved for future use

5.4 Additional key layers

5.4.1 Overview

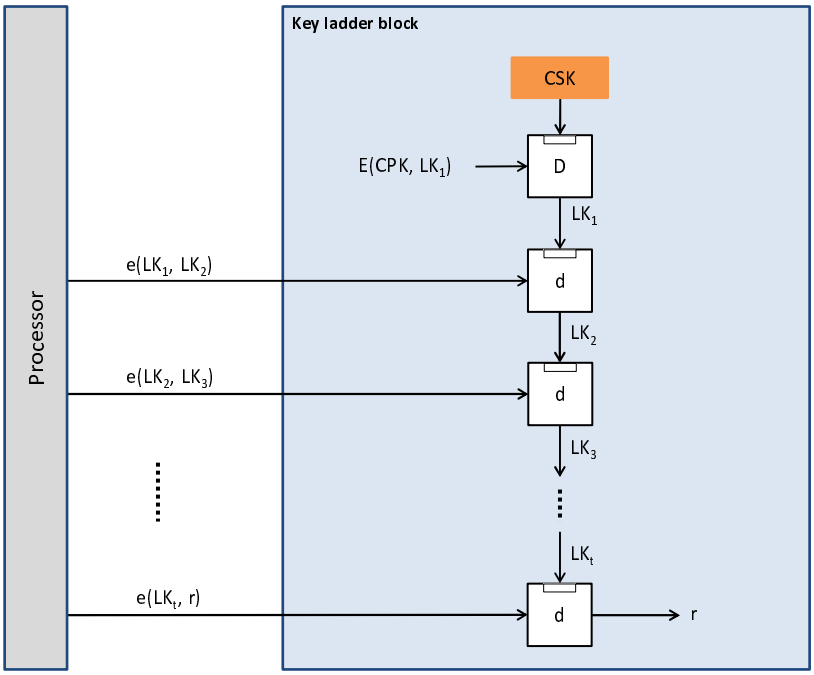

Figure 5.4.1-1: Additional key layers

The specification in clauses 5.1 and 5.2 assumes that one key layer of the key ladder is associated with link keys. The key ladder shall also support additional key layers for link keys, as depicted in Figure 5.4.1-1 (only the building blocks of the key ladder that are relevant to the discussion in this clause are depicted in the figure).

As in Figure 5.4.1-1, let t denote the number of link keys in the key ladder. The key ladder block shall support key ladders for all values of t with $1 \leq t \leq 24$. Note that $t = 1$ in the scheme specified in clauses 5.1 and 5.2.

5.4.2 Key ladder computations

The sender can generate the t input messages to the key ladder block using the following steps.

Compute $e(LK_1, LK_2)$, $e(LK_2, LK_3)$, … , $e(LK_{t-1}, LK_t)$, and $e(LK_t, r)$ (sender):

1) Generate link keys LK_i for i = 1, 2, …, t and a random bit string r.

2) Compute $e(LK_{i-1}, LK_i)$ for i = 2, 3, ..., t.

3) Compute $e(LK_t, r)$.

After receiving $e(LK_1, LK_2)$, $e(LK_2, LK_3)$, … , $e(LK_{t-1}, LK_t)$, and $e(LK_t, r)$, the key ladder block shall compute r using the following steps (see also Figure 5.4.1-1).

Compute r (key ladder block):

1) Compute $LK_i = d(LK_{i-1}, e(LK_{i-1}, LK_i))$ for i = 2, 3, ..., t.

2) Compute $r = d(LK_t, e(LK_t, r))$.

5.5 Associated Data 2

The sender can optionally send Associated Data 2, also referred to as AD2, together with $e(LK_t, r)$ to the key ladder. The value of the bit string τ_b (as introduced in clause 5.1) signals the presence of AD2. As specified in clause 5.1, the length of τ_b is 8 bits. In the following text, τ denotes the integer representation of τ_b in which the least significant bit of τ corresponds to the rightmost bit of τ_b. The key ladder shall support all values of $\tau \in \{0, 64, 96, 128\}$. If $\tau = 0$, then AD2 shall not be present, and the scheme as presented in clauses 5.1 to 5.4 shall be used. The description below assumes that $\tau \neq 0$. In this case AD2 shall be present. The key ladder block shall verify if the received $\tau \in \{64, 96, 128\}$; if this does not hold true, then the key ladder block shall abort the computations.

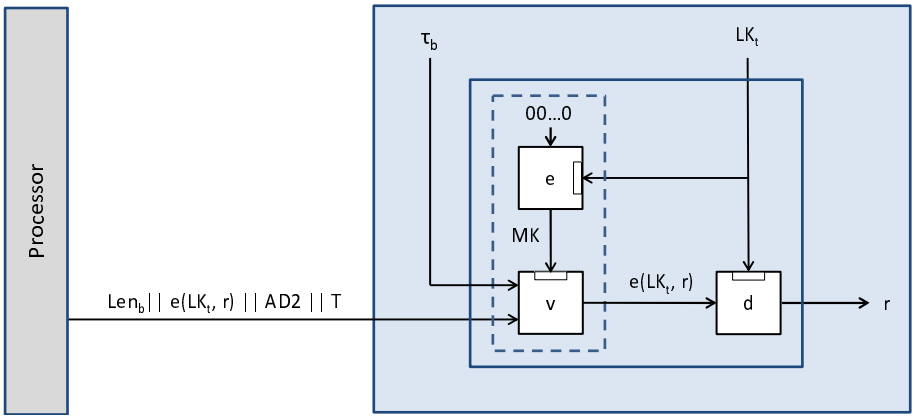

Figure 5.5-1: Associated Data 2

The present document assumes that AD2 is represented as a bit string. Let Len denote the bit length of AD2. The key ladder block shall support all values of Len with $1 \leq \text{Len} \leq 256$.

Let Len_b denote the binary representation of the integer Len - 1 in which the rightmost bit of Len_b corresponds to the least significant bit of Len - 1. The length of Len_b shall be 8 bits.

The bit string AD2 shall be cryptographically bound to the corresponding ciphertext $e(LK_t, r)$ during its transport to the key ladder. To this end, the sender and the key ladder block shall use a **message authentication code algorithm**, denoted by mac. The inputs to this algorithm are the secret MAC key MK, the message $\text{Len}_b \,||\, e(LK_t, r) \,||\, \text{AD2}$, and τ. The output of mac is a tag T with a length of τ bits. The algorithm mac is specified in clause 8.5.

Let 00…0 denote the all-zero bit string of 128 bits. The sender can use the following steps to generate the input message $\text{Len}_b \,||\, e(LK_t, r) \,||\, \text{AD2} \,||\, T$ to the key ladder (as depicted in Figure 5.5-1).

Compute $\text{Len}_b \,||\, e(LK_t, r) \,||\, \text{AD2} \,||\, T$ (sender):

1) Compute $MK = e(LK_t, 00\ldots0)$.

2) Prepend Len_b to the bit string $e(LK_t, r) \,||\, \text{AD2}$.

3) Compute $T = \text{mac}(MK, \text{Len}_b \,||\, e(LK_t, r) \,||\, \text{AD2}, \tau)$.

4) Append T to the bit string $\text{Len}_b \,||\, e(LK_t, r) \,||\, \text{AD2}$.

The message Len_b || $e(LK_t, r)$ || AD2 || T replaces the message $e(LK_t, r)$ that is used if $\tau = 0$. After receiving Len_b || $e(LK_t, r)$ || AD2 || T, the key ladder block shall compute $e(LK_t, r)$ using the following steps (Steps 2 - 4 are denoted by v in Figure 5.5-1).

Compute $e(LK_t, r)$ (key ladder block):

1) Compute $MK = e(LK_t, 00…0)$.

2) Compute $T = mac(MK, Len_b || e(LK_t, r) || AD2, \tau)$.

3) Verify if the received T equals the computed T. If these two values are not equal, then the key ladder block shall abort the computations.

4) Retrieve $e(LK_t, r)$ from the received message.

The specification of the contents of AD2 is outside the scope of the present document and the present document assumes that the key ladder block does not process AD2 other than using it as input to the algorithm mac. The key ladder block may pass AD2, or part of AD2, together with CW-URI and CW to the **content descrambler**.

If $\tau \neq 0$, then the implementation of the key ladder shall ensure that the value of τ_b that was provided together with Len_b || $e(LK_t, r)$ || AD2 as input to the algorithm mac is also provided as input to h when CW is derived from r.

Only the key ladder block shall have access to a plaintext CSK, LK_i ($1 \leq i \leq t$), MK, and r.

6 Authentication mechanism

6.1 Overview

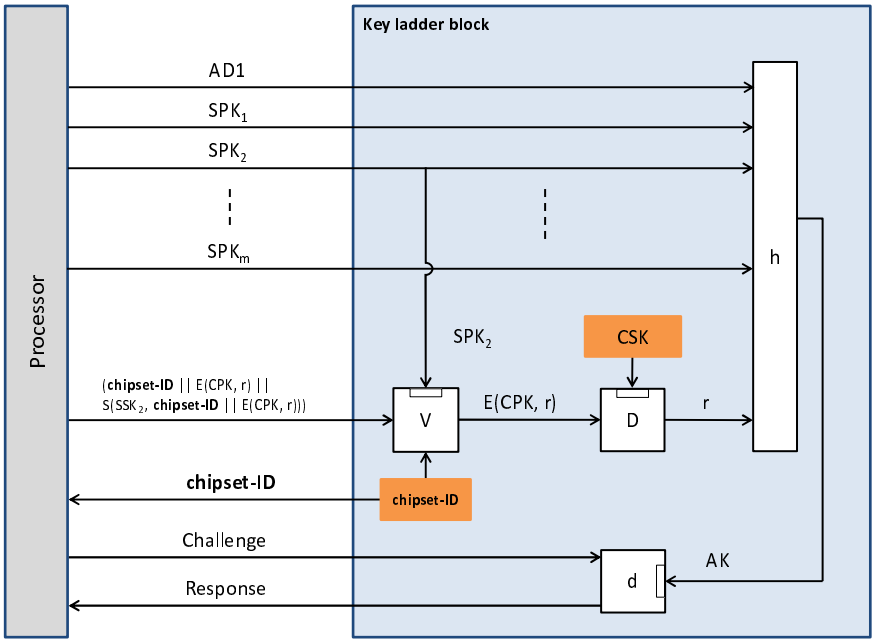

Figure 6.1-1: Authentication mechanism

This clause presents the functional design of the authentication mechanism in the key ladder block. The authentication mechanism is depicted in Figure 6.1-1. This mechanism is closely related to the key ladder as described in clause 5; in particular, it uses the same operations V, D and function h, and it uses the same **chipset-ID** and CSK. However, the authentication mechanism does not make use of link keys and CW-URI, SPK-URI and τ_b are not inputs to the authentication mechanism and the function h. Moreover, the output of h is not a **control word** CW used for content decryption or content encryption (in particular, the authentication mechanism does not interface directly with the **content descrambler**), but an authentication key AK. AK is a key of the same **symmetric encryption scheme** as used in the key ladder. As shown in Figure 6.1-1, only its decryption operation d is used in the implementation of the authentication mechanism.

The input message (**chipset-ID** || E(CPK, r) || S(SSK_i, **chipset-ID** || E(CPK, r))) is also denoted by SIM_{AUTH} in the following text.

6.2 Authentication mechanism computations

The sender associated with key pair (SSK_i ,SPK_i) for some $1 \leq i \leq m$ can create the signed input message SIM_{AUTH}, i.e.

(**chipset-ID** || E(CPK, r) || S(SSK_i, **chipset-ID** || E(CPK, r))),

using the following steps.

Compute SIM_{AUTH} (sender):

1) Generate a random bit string r.

2) Compute the ciphertext E(CPK, r).

3) Concatenate **chipset-ID** and E(CPK, r); the resulting bit string is denoted by (**chipset-ID** || E(CPK, r)).

4) Sign the message (**chipset-ID** || E(CPK, r)) using SSK_i; the signature is denoted by S(SSK_i, **chipset-ID** || E(CPK, r)).

5) Append this signature to the bit string (**chipset-ID** || E(CPK, r)).

After receiving SIM_{AUTH} and sender public key SPK_i, the key ladder block shall perform the following steps to retrieve r (in Figure 6.1-1, the first two steps are denoted by V).

Compute r (key ladder block):

1) Verify if the received **chipset-ID** equals the stored **chipset-ID**. If these two values are not equal, then the key ladder block shall abort the computations.

2) Use the received SIM_{AUTH} and SPK_i to verify the signature. If the signature is invalid, then the key ladder block shall abort the computations.

3) Compute r = D(CSK, E(CPK, r)).

In case of the authentication mechanism, V is allowed to use any SPK_i with $1 \leq i \leq m$ to verify the signature of SIM_{AUTH} (in Figure 6.1-1, it is assumed that i = 2).

Next, AD1, SPK_1, SPK_2, ..., SPK_m and r shall be provided as inputs to the function h. The output of h is the authentication key AK. The implementation of the authentication mechanism shall ensure that the public key that was used to verify the authenticity of SIM_{AUTH} (or more precisely, the SIM_{AUTH} containing the random number r) is provided as one of the SPK-inputs to h when AK is derived from r.

Next, the sender can create an input message referred to as Challenge. After receiving Challenge from the processor, the authentication mechanism shall compute Response using the following step.

Compute Response (key ladder block):

1) Compute Response = d(AK, Challenge).

As depicted in Figure 6.1-1, the authentication mechanism shall return Response to the processor.

Only the key ladder block shall have access to a plaintext AK.

7 Data conversion primitives

7.1 BS2OSP

The bit string to octet string primitive BS2OSP is used to define the **public-key encryption scheme** in clause 8.2 and the **digital signature scheme** in clause 8.3.

Function: BS2OSP(x)

Input: x a bit string of length 8j ($j \geq 1$).

Output: X an octet string of length j.

Steps:

1) Let x_i denote a bit for $0 \leq i \leq 8j-1$, and let $x = x_0 x_1 \ldots x_{8j-1}$.

2) Let X_i be the octet defined by $X_i = x_{8i+1} \ldots x_{8i+7}$ for $0 \leq i \leq j-1$.

3) Output the octet string $X = X_0 X_1 \ldots X_{j-1}$.

7.2 OS2BSP

The octet string to bit string primitive OS2BSP is used to define the I2BSP primitive in clause 7.3 and the **public-key encryption scheme** in clause 8.2.

Function: OS2BSP(X)

Input: X an octet string of length j ($j \geq 1$).

Output: x a bit string of length 8j.

Steps:

1) Let X_i denote an octet for $0 \leq i \leq j-1$ and let $X = X_0 X_1 \ldots X_{j-1}$.

2) Let x_i denote a bit for $0 \leq i \leq 8j-1$ and let these bits be defined as $X_i = x_{8i+1} \ldots x_{8i+7}$ for $0 \leq i \leq j-1$.

3) Output the bit string $x = x_0 x_1 \ldots x_{8j-1}$.

7.3 I2BSP

The integer to bit string primitive I2BSP is used to define the function h in clause 8.4.

Function: I2BSP(x)

Input: x a 2 048-bit integer.

Output: a bit string of length 2 048.

Step: If the integer to octet string data conversion primitive I2OSP(x, xLen) is defined as in [2], then I2BSP(x) = OS2BSP(I2OSP(x, 256)).

8 Cryptographic operations

8.1 Symmetric encryption scheme

The **symmetric encryption scheme** used in the key ladder block shall be AES-128 [3] in ECB mode of operation.

8.2 Public-key encryption scheme

The **public-key encryption scheme** shall be RSAES-PKCS1-v1_5 [2]. This scheme consists of an encryption operation and a decryption operation. Recall from clauses 5 and 6 that only the decryption operation is implemented in the key ladder block, and that the sender's system implements the encryption operation.

The chipset public key CPK shall be equal to the recipient's RSA public key (*n*, *e*) in [2]. In the present document the bit length of *n* shall be 2 048, i.e. the octet length of *n*, denoted by *k* in [2], equals 256. The secret/private key CSK shall be equal to the recipient's RSA private key *K* in [2]. The representation of CSK shall be equal to one of the representations of *K* specified in [2].

The message to be encrypted is denoted by *M* in [2] and represented as an octet string. In the present document, *M* is defined as *M* = BS2OSP(LK) in case of the key ladder, and *M* = BS2OSP(r) in case of the authentication mechanism. As a result, the octet length of *M* equals 16 in the present document.

CPK and *M* are input to the encryption operation:

RSAES-PKCS1-v1_5-Encrypt(CPK, *M*).

The output of this operation is a ciphertext, denoted by *C* and represented as an octet string in [2]. The octet length of *C* equals 256 in the present document. *C* equals BS2OSP(E(CPK, LK)) in case of the key ladder and BS2OSP(E(CPK, r)) in case of the authentication mechanism.

CSK and *C* are input to the decryption operation:

RSAES-PKCS-v1_5-Decrypt(CSK, *C*).

This operation is implemented in the key ladder block. The output of this operation is the message *M*, represented as an octet string. Next, the key ladder block shall compute LK = OS2BSP(*M*) in case of the key ladder and r = OS2BSP(*M*) in case of the authentication mechanism.

8.3 Digital signature scheme

The **digital signature scheme** shall be RSASSA-PKCS1-v1_5 [2]. This scheme consists of a signature generation operation and a signature verification operation. Recall from clauses 5 and 6 that only the signature verification operation is implemented in the key ladder block, and that the sender's system implements the signature generation operation.

The signer's RSA public key is denoted by (*n*, *e*) and represented as a pair of integers in [2]. In the present document, the length of the modulus *n* of the signer's RSA public key shall be 2 048 bits, i.e. the octet length of *n*, denoted by *k* in [2], equals 256. Further, the value of the public exponent *e* shall be equal to $2^{16} + 1$ in the present document. A compliant chipset shall permanently store this value and the chipset shall implement measures to protect the integrity of this value. The sender public key SPK shall be equal to the modulus *n*, implying that the length of SPK equals 2 048 bits and that SPK is represented as an integer. The sender secret/private key SSK shall be equal to the signer's RSA private key *K* in [2].

The message to be signed is denoted by *M* and represented as an octet string in [2]. In the present document, *M* is defined as:

M = BS2OSP(**chipset-ID** || E(CPK, LK))

in case of the key ladder; and

M = BS2OSP(**chipset-ID** || E(CPK, r))

in case of the authentication mechanism. As a result, the octet length of *M* is equal to 8 + 256 = 264 in both cases.

SSK and *M* are input to the signature generation operation:

RSASSA-PKCS1-v1_5-Sign(SSK, *M*).

The output of the signature generation operation is a signature, denoted by *S* and represented as an octet string in [2]. The octet length of *S* equals 256 in the present document. *S* equals BS2OSP(S(SSK_i, **chipset-ID** || E(CPK, LK)))) in case of the key ladder and BS2OSP(S(SSK_i, **chipset-ID** || E(CPK, r)))) in case of the authentication mechanism.

(SPK, 2^{16} + 1), *M* and *S* are input to the signature verification operation:

RSAES-PKCS-v1_5-Verify((SPK, 2^{16} + 1), *M*, *S*).

This operation is implemented in the key ladder block. The output of this operation is either "valid signature" or "invalid signature".

8.4 Function h

This clause specifies the function h. Recall from clauses 5 and 6 that the inputs to h are

- CW-URI, τ_b, AD1, SPK-URI, SPK_1, SPK_2, ..., SPK_m, and r in case of the key ladder.
- AD1, SPK_1, SPK_2, ..., SPK_m, and r in case of the authentication mechanism.

If the key ladder block does not receive one of these two sets of inputs, or if the length of an input is not as specified in the present document, then the key ladder block shall abort the computations. Otherwise, h shall first apply the I2BSP data conversion primitive to each of its SPK inputs. Next, h shall concatenate the bit stings representing its inputs as follows to obtain the message *M*. In case of the key ladder:

M = r || CW-URI || τ_b || AD1 || SPK-URI || I2BSP(SPK_1) || I2BSP(SPK_2) || ... || I2BSP(SPK_m),

and in case of the authentication mechanism:

M = r || AD1 || I2BSP(SPK_1) || I2BSP(SPK_2) || ... || I2BSP(SPK_m).

Recall that the length of r, CW-URI, τ_b, AD1, SPK-URI and I2BSP(SPK_i) is 128 bits, 64 bits, 8 bits, 256 bits, 64 bits, and 2 048 bits, respectively. As a result, the bit-length of *M*, denoted by *l* in [4], equals 520 + 2 048 m in case of the key ladder and 384 + 2 048m in case of the authentication mechanism.

Next, h shall compute SHA-256(*M*) as defined in [4].

In case of the authentication mechanism, h shall truncate this 256-bit message digest to 128 bits and it shall output this truncated message digest. In case of the key ladder, the key ladder block shall pass the 256-bit output of SHA-256 to the **content descrambler**, and if the length of CW is N bits, then the **content descrambler** shall truncate this output to N bits. In both cases, the truncation method as defined in [5] shall be used.

8.5 Message authentication code algorithm

This clause defines the algorithm mac as introduced in clause 5.5. This **message authentication code algorithm** is defined as in [6] with the following selections:

- The block cipher shall be AES-128.
- Padding of the message shall be done with Padding Method 3. This method needs the bit length of the unpadded message e(LK_t, r) || A. This bit length equals Len + 129.
- MAC Algorithm 1 shall be used to compute the MAC from the message and the secret key.
- The length of the MAC shall be τ bits.

History

Document history		
V1.1.1	July 2017	Publication