International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

J.1003

(10/2014)

SERIES J: CABLE NETWORKS AND TRANSMISSION OF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS

Conditional access and protection – Renewable conditional access system

Specifications of network protocol for renewable conditional access system

Recommendation ITU-T J.1003

Recommendation ITU-T J.1003

Specifications of network protocol for renewable conditional access system

Summary

Recommendation ITU-T J.1003 specifies the renewable conditional access system (RCAS) network protocol that supports authentication and secure download of conditional access client software (CACS) required in Recommendation ITU-T J.1001.

History

Edition	Recommendation	Approval	Study Group	Unique ID*
1.0	ITU-T J.1003	2014-10-29	9	11.1002/1000/12323

[*] To access the Recommendation, type the URL http://handle.itu.int/ in the address field of your web browser, followed by the Recommendation's unique ID. For example, http://handle.itu.int/11.1002/1000/11830-en.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure, e.g., interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

© ITU 2015

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

Table of Contents

Recommendation ITU-T J.1003

Specifications of network protocol for renewable conditional access system

1 Scope

This Recommendation specifies the renewable conditional access system (RCAS) network protocol that supports authentication and secure download of conditional access client software (CACS) required in [ITU-T J.1001].

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T J.1001] Recommendation ITU-T J.1001 (2012), *Requirements for renewable conditional access system*.

[ITU-T J.1002] Recommendation ITU-T J.1002 (2013), *Pairing protocol specification for renewable conditional access system*.

[ITU-T X.509] Recommendation ITU-T X.509 (2008) | ISO/IEC 9594-8:2008, *Information technology – Open systems interconnection – The Directory: Public-key and attribute certificate frameworks*.

[ITU-T X.690] Recommendation ITU-T X.690 (2008) | 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).*

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 conditional access (CA) [b-ITU-T J.193]: The conditional granting of access to cable services and content based upon what service suite has been purchased by the customer.

3.1.2 descrambling [b-ITU-T J.93]: The processes of reversing the scrambling function (see "scrambling") to yield usable pictures, sound and data services.

3.1.3 entitlement control messages (ECMs) [b-ITU-T J.290]: An ECM is an encrypted message that contains access criteria to various service tiers and a control word (CW).

3.1.4 entitlement management messages (EMMs) [b-ITU-T J.290]: The EMM contains the actual authorization data and shall be sent in a secure method to each CPE device.

3.1.5 scrambling [b-ITU-T J.93]: The process of using an encryption function to render television and data signals unusable to unauthorized parties.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 Security symbols

Security symbols	Descriptions
Pub(X)	Public key of X
Prv(X)	Private key of X
E(k,m)	Encryption of a message m with key k.
S(k,m)	Digital signature for a message m with signing key k.
H(m)	SHA-256 hashing for a message m
HMAC(k,m)	HMAC-SHA1 for a message m with key k
X||Y	Concatenation of X and Y
Cert(X)	X.509 certificate of X
PRF(X)	Pseudo random function having a seed value of X
$X_{msb(Y)}$	Y bits from MSB of X

3.2.2 Parameter definitions

Parameter names	Descriptions
DSC_ID	The value of identification of DSC having a size of 40 bytes
CAM_ID	The value of identification of CAM having a size of 8 bytes
KeyPairingID	The value of concatenation with CAM_ID and DSC_ID, i.e., CAM_ID||DSC_ID
CWEK	The abbreviation of control words encryption key and used to encrypt *control words*. The CWEK generation method is CWEK = H(CWEK||CAM_ID||DSC_ID)msb(128)
KPK	The abbreviation of key pairing key. The authorization centre (AC) generates the KPK if the KeyPairingID is valid.
HMAC_KEY	A hash-based message authentication code (HMAC) secret key. CAM uses HMAC_KEY to generate an HMAC value for the message including *control words*. The HMAC_KEY generation method is HMAC_KEY = H($RAND_{HMAC}$||CAM_ID||DSC_ID)msb(160), Here $RAND_{HMAC}$ is achieved by $PRF(X)_{msb(320)}$
RAND	Random number with 320 bits
K_i	The pre-shared key having the size of 128 bits. The AC uniquely assigns three K_i to each CAM. K_i should be a generated random generation function.

4 Abbreviations and acronyms

This Recommendation uses the following Abbreviations and acronyms:

AC	Authorization Centre
ASN	Abstract Syntax Notation
BER	Basic Encoding Rules
CACS	Conditional Access Client Software

CAM	Conditional Access Module
CASS	CAM Authentication Sub-System
CCCIEK	Common CAM Client Image Encryption Key
CCI	CAM Client Image
CHK	Common Hash Key
CPE	Customer Premise Equipment
CWEK	Control Words Encryption Key
DOCSIS	Data Over Cable Service Interface Specification
HMAC	Hash-based Message Authentication Code
ICCIEK	Individual CAM Client Image Encryption Key
IHK	Individual Hash Key
IV	Initialization Vector
KPK	Key Pairing Key
MEK	Message Encryption Key
MK	Master Key
PRF	Pseudo Random number generation Function
PRNG	Pseudo Random Number Generator
RCAS	Renewable Conditional Access System

5 Conventions

In this Recommendation:

The keywords "**is required to**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

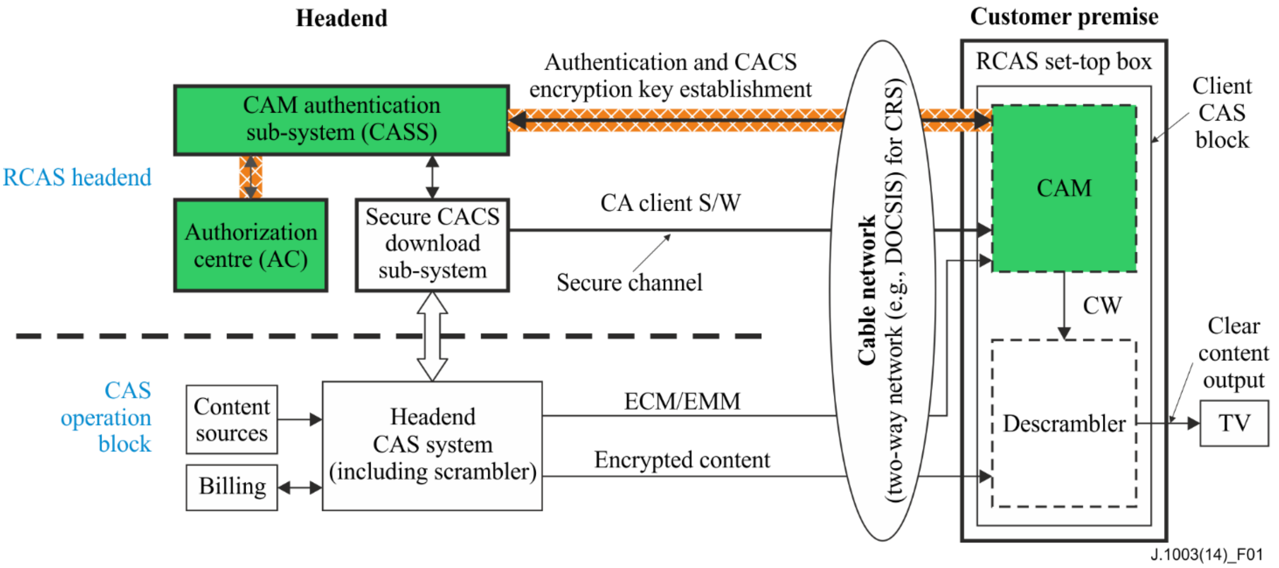
The keywords "**is recommended**" indicate a requirement which is recommended but which is not absolutely required. Thus this requirement need not be present to claim conformance.

The keywords "**is prohibited from**" indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords "**can optionally**" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

In the body of this document and its annexes, the words *shall*, *shall not*, *should*, and *may* sometimes appear, in which case they are to be interpreted, respectively, as *is required to*, *is prohibited from*, *is recommended*, and *can optionally*. The appearance of such phrases or keywords in an appendix or in material explicitly marked as *informative* are to be interpreted as having no normative intent.

6 Overview of RCAS network protocol



Headend
Customer premise
RCAS headend
CAM authentication sub-system (CASS)
Authentication and CACS encryption key establishment
Authorization centre (AC)
Secure CACS download sub-system
CA client S/W
Secure channel
Cable network (two-way network (e.g., DOCSIS) for CRS)
RCAS set-top box
Client CAS block
CAM
CW
Clear content output
Descrambler
TV
CAS operation block
Content sources
Billing
Headend CAS system (including scrambler)
ECM/EMM
Encrypted content
J.1003(14)_F01

Figure 1 – Reference architecture of RCAS

Figure 1 is a reference architecture illustrating a configuration of a renewable conditional access system (RCAS). The RCAS provides a mutual authentication method between a conditional access module (CAM) and a CAM authentication sub-system (CASS). A mutual authentication includes the CAM of a RCAS set-top box, the CASS of a RCAS headend and an authorization centre (AC) connected to the CASS. As shown in Figure 1 the CAM and the CASS interactively communicate with each other through a cable network for this mutual authentication. The AC provides a variety of important information used for authentication through the CASS. The CASS transmits information used for authentication received from the AC to the CAM by means of a suitable communication system such as data over cable service interface specification (DOCSIS). All types of key information generated during the authentication are managed by the AC. When the authentication is completed normally, the conditional access client software (CACS) is transmitted to the CAM through a secure CACS download sub-system. After downloading the CACS, the CAM obtains viewing entitlement with respect to a scrambled and transmitted broadcasting signal and provides a subscriber with a fee-based broadcasting service through customer premise equipment (CPE). The messages exchanged among the CAM, the CASS and the AC are defined as a RCAS network protocol. The RCAS network protocol enables a security and authentication function for messages exchanged among the CAM, the CASS and the AC.

The specification of RCAS network protocol includes the following functions for authentication and secure download of CACS that are specified in [ITU-T J.1001]:

- Security announcement function: The ability to provide information such as initial configuration, upgrade, or access information, to CAMs in the network through the RCAS network protocol.
- Key establishment and authentication function: The establishment CACS encryption and authentication key which is used for encrypting CACS to protect the image of CACS while it is delivered from the headend to CAM.
- Supporting RCAS pairing function: It is required that RCAS network protocol supports the RCAS pairing function which is specified in [ITU-T J.1002] while RCAS network protocol is operated.
- Renewal information announcement function: The ability to provide the CACS download information to CAM through the RCAS network protocol.

7 Protocol processing procedures

Figure 2 is a sequence chart illustrating a mutual authentication method based on RCAS network protocol. In this protocol, it is assumed that the CAM retains an AC certificate (AC X.509 certificate), a CAM certificate, a Ki value and a three operator variant algorithm configuration field (OP). The CASS is assumed to retain an AC certificate (AC X.509 certificate) and a CASS certificate (CASS X.509 certificate). The AC is assumed to retain an AC certificate (AC X.509 certificate), a CASS certificate (CASS X.509 certificate), a CAM certificate, a three OP, a Ki value and a key pairing identifier (ID). Under these assumptions, the sequence chart of a mutual authentication method includes an 'Announcement phase', a 'Key establishment phase', a 'CAM registration phase' and a 'CACS renewal phase' as shown in the Figure 2.

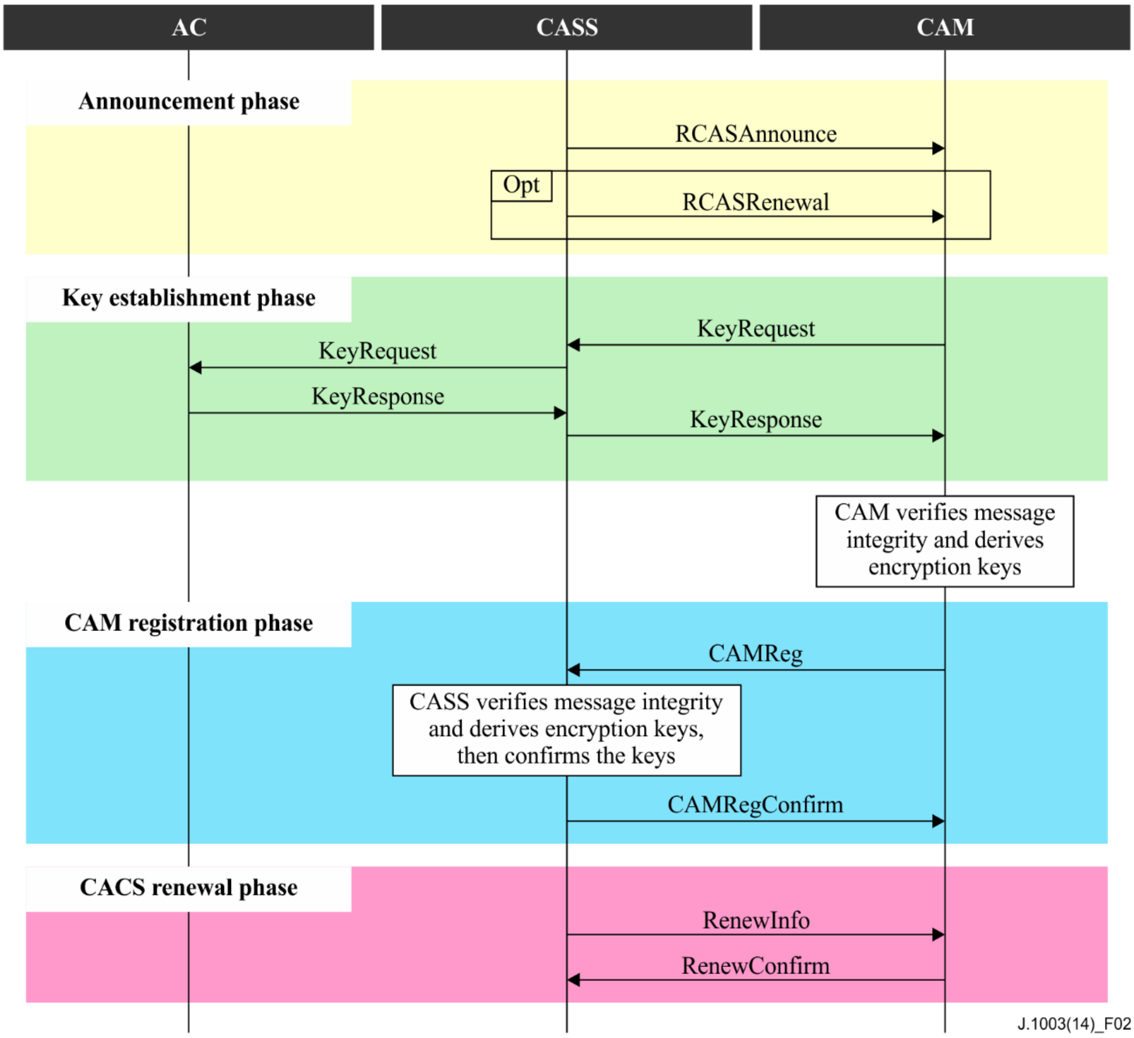


Figure 2 – A sequence chart of RCAS network protocol

7.1 Announcement phase

In the announcement phase, the RCAS headend controls the CASS to transmit an RCASAnnounce message or an RCASRenewal message to the CAM. The role of this message is to notify the start of RCAS network protocol as well as to provide the CASS server access information to all CAMs.

The RCASAnnounce message provides the 'CAM Client Version’ information such as 'CAM HW version', 'CAM SW version' and 'CACS type', etc., as well as 'CASS' information such as the 'IP address of CASS', 'IP connection type (e.g., TCP or UDP)' and the 'Listening port number', etc. The RCASAnnounce message is authenticated by the CASS using a hash-based message authentication code (HMAC) scheme and the CASS transmits the authenticated RCASAnnounce message to the

CAM using a multicast scheme. The CAM performs a HMAC message authentication using a common hash key (CHK) which is assumed to be included in the CAM before the start of this phase. The HMAC message authentication is performed to authenticate the RCASAnnounce message received from the CASS and accordingly the CAM performs a key establishment phase. It must be noted that CAM should acquire CHK from the key establishment phase when the CHK of the CAM differs from that of the CASS, or when the CAM is moved to a CASS zone, or when the CAM is in a virgin state.

The RCAS headend optionally can send an RCASRenewal message to CAMs using CASS when the RCAS headend wants to spread the download time intentionally as well as request key updates or purchase information. By utilizing this message, the RCAS headend can let a CAM know the CACS download time, or request the CAM to send a KeyRequest message or purchase information. To do this, the RCASRenewal message provides the information of CACS image download time, key upgrade request and purchase information request, etc. The RCASRenewal message is authenticated using the same method as that for the RCASAnnounce message.

7.2 Key establishment phase

In the key establishment phase, the RCAS headend receives a KeyRequest message from the CAM using the CASS in response to the RCASAnnounce message and transmits the received KeyRequest message to the AC. CASS receives the KeyResponse information from the AC in response to the KeyRequest message and transmits the received KeyResponse message to the CAM.

Specifically, CAM transmits, to the CASS, a KeyRequest message digitally signed by a private key of the CAM.

The RCAS headend verifies a digital signature of the KeyRequest message using the CASS and transmits a new KeyRequest message to the AC. Here, the new KeyRequest message is regenerated based on a key pairing ID and a CASS ID extracted from the KeyRequest message.

The RCAS headend searches for a CAM certificate based on the key pairing ID using the AC and authenticates the CAM based on the CAM certificate. The AC defines a result of the authentication of the CAM in the KeyResponse message and transmits the KeyResponse message to the CASS.

When the CAM is in the virgin state, the AC performs a transfer protocol pairing function. Alternatively, when the CAM is not in the virgin state, the AC performs a function of comparing the received pairing information with an initial pairing value. The detail specification of this pairing process is defined in [ITU-T J.1002].

Along with verifying the validation of CAM, the AC performs the process of CAM cloning detection. The detail descriptions concerning CAM cloning detection process are included in Annex C.

The RCAS headend defines a CASS certificate in the KeyResponse message using the CASS and transmits the KeyResponse message to the CAM.

When an authentication result value (Auth_Rst) about the KeyResponse message is set as true, the CASS generates a CHK and an individual hash key (IHK) through a hash key generation process. Then CASS adds the generated CHK and IHK together with the CASS certificate to the KeyResponse message. In addition, the CASS digitally signs the KeyResponse message using a private key of the CASS and encrypts a part of the digitally signed KeyResponse message using a public key of the CAM. Finally CASS transmits the encrypted KeyResponse message to the CAM.

On the CAM side, a decryption unit of the mutual authentication decrypts the KeyResponse message. The decryption unit decrypts one or more pieces of information contained in the KeyResponse message based on the CAM certificate.

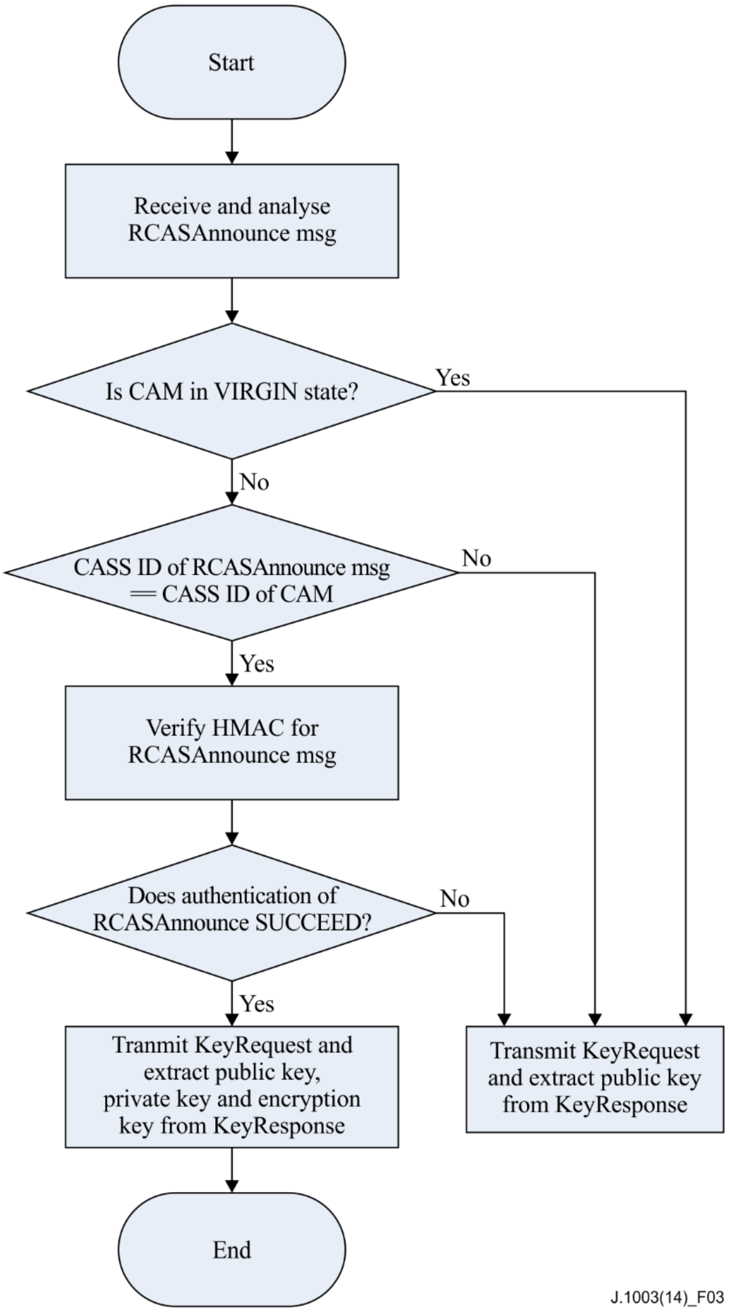

Start
Receive and analyse RCASAnnounce msg
Is CAM in VIRGIN state?
Yes
No
CASS ID of RCASAnnounce msg == CASS ID of CAM
No
Yes
Verify HMAC for RCASAnnounce msg
Does authentication of RCASAnnounce SUCCEED?
No
Yes
Tranmit KeyRequest and extract public key, private key and encryption key from KeyResponse
Transmit KeyRequest and extract public key from KeyResponse
End
J.1003(14)_F03

Figure 3 – Flow chart for Announcement & Key establishment phases in CAM

Figure 3 is a flowchart for the processes of announcement and key establishment phases in the CAM. The CAM receives the RCASAnnounce message and analyses the received RCASAnnounce message. The CAM also determines whether a current state is in the virgin state. When the CAM is in the virgin state or when the CAM is moved to the CASS zone, the updating unit of the decryption unit extracts a newest CHK and updates the original CHK. The CAM determines whether a CASS ID contained in the RCASAnnounce message is identical to a CASS ID contained in the CAM. When determining that the two CASS IDs are different, the CAM performs the 'Transmit KeyRequest & extract public key from KeyResponse' operation. However, when the CAM is not in the virgin state, or when the CAM is not moved to the CASS zone, the authentication unit of the decryption unit performs the HMAC message authentication using the CHK retained in the CAM. The CAM also determines whether authentication of the RCASAnnounce message succeeds. When the authentication of the RCASAnnounce message is determined to have failed, the CAM performs the 'Transmit KeyRequest & extract public key from KeyResponse' operation. Alternatively, when the authentication of the RCASAnnounce message is determined to have succeeded, the CAM transmits the KeyRequest message to the CASS and extracts a public key, a private key and an encryption key from the KeyResponse message.

7.3 CAM registration phase

Between the key establishment phase and the CAM registration phase, an encryption key generation process takes place in the CASS using the method shown in Figure 4. The CASS generates two different encryption keys, the message encryption key (MEK) and the individual CAM client image encryption key (ICCIEK). The MEK and ICCIEK have a key length of 128 bits, are generated using an input of a pseudo random number generator (PRNG) as a master key (MK). Three K_C values among input values of the SHA-1 hash function means that three K_C are generated using three RAND values in RAND_AC received from a CASS through a KeyResponse message.

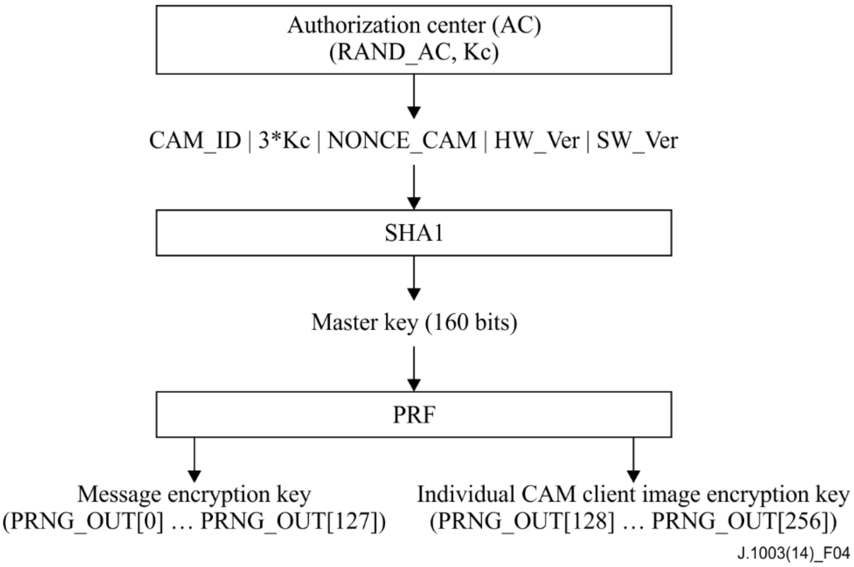

Authorization center (AC)
(RAND_AC, Kc)
CAM_ID | 3*Kc | NONCE_CAM | HW_Ver | SW_Ver
SHA1
Master key (160 bits)
PRF
Message encryption key
(PRNG_OUT[0] … PRNG_OUT[127])
Individual CAM client image encryption key
(PRNG_OUT[128] … PRNG_OUT[256])
J.1003(14)_F04

Figure 4 – Encryption keys generation method

In the CAM registration phase, the CAM transmits to the CASS, a CAMReg message containing hashed encryption keys which are the MEK and ICCIEK. After receiving a CAMReg message, the CASS generates two encryption keys using the same method shown in Figure 4 and then determines whether the encryption keys from the CAM are identical to the encryption keys which were generated in the CASS by comparing the hashed values of the MEK and ICCIEK. Then the CASS controls the CAMRegConfirm message to be transmitted to the CAM in response to the CAMReg message when the encryption keys are determined to be identical.

When the encryption keys from the CAM are determined to differ from that from the CASS, the CASS transmits inconsistency information through a status message.

The CAMRegConfirm message is authenticated using the HMAC-SHA1 method with the IHK and is encrypted using AES, the common CAM client image encryption key (CCCIEK), the initialization vector (IV) for ICCIEK encryption, the initialization vector for CCCIEK encryption, the initialization vector for MEK encryption and the Session_ID. The CCCIEK generation method is shown in Figure 5.

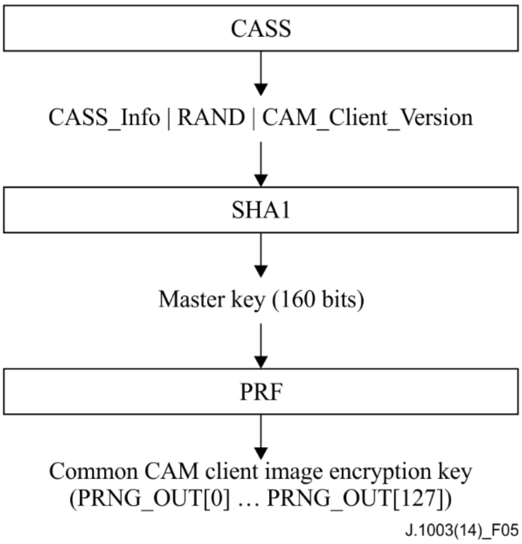

CASS
CASS_Info | RAND | CAM_Client_Version
SHA1
Master key (160 bits)
PRF
Common CAM client image encryption key
(PRNG_OUT[0] … PRNG_OUT[127])
J.1003(14)_F05

Figure 5 – CCCIEK generation method

7.4 CACS renewal phase

In the CACS renewal phase, the RCAS headend controls RenewInfo from the CASS to the CAM. Here, the RenewInfo is used to permit the CAM to download CACS image information.

After the HMAC message authentication is performed using the private key and a message is encrypted using the AES algorithm with the MEK and the IV of the MEK, the RenewInfo is transmitted to the CAM.

The CAM receives the RenewInfo and normally performs message authentication and decryption operations. It then downloads the CACS image information from a server in which the CACS image information is stored.

Since the CACS image information is encrypted using the AES algorithm with the ICCIEK and the CCCIEK, the CAM decrypts the CACS image information using the ICCIEK and the CCCIEK.

A RenewConfirm message in response to the RenewInfo is transmitted from the CAM to the CASS. In addition when the PurchaseReport_REQ is defined in the RenewInfo, the CAM applies the HMAC to the PurchaseReport message using the encryption key. Then the CAM transmits the encrypted PurchaseReport message to the CASS.

8 Protocol security

8.1 Message security

In the cases of KeyRequest and KeyResponse messages, the message header and message content are digitally signed and delivered with the message itself. Some parameters in the KeyResponse message are encrypted with the public key encryption method. The other messages except the KeyRequest and KeyResponse are encrypted and authenticated as shown in Figure 6. The AES-128-CBC method is used for message content encryption. Note that message contents are basic encoding rules (BER) encoded before being AES-128-CBC encrypted. For message authentication and integrity, HMAC-SHA1 is used. The inputs of the HMAC-SHA1 process are the message header and the BER encoded message content.

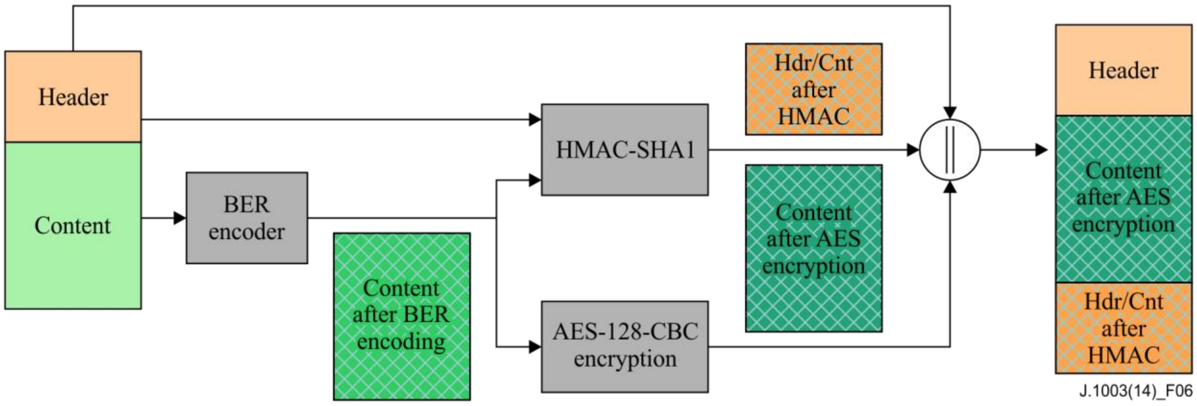


Figure 6 – Message encryption and the HMAC process in an RCAS headend

8.2 CAM client image security

The CAM client image is encrypted using the AES-128-CBC method. For CAM client image integrity, the SHA1 algorithm is applied to the CAM client image itself and the RCAS headend delivers the output of the SHA1 algorithm to a CAM using the RenewInfo message. On the CAM side, the decryption and integrity check process are carried out as shown in Figure 7. Initially the CAM client image is decrypted using the AES-128-CBC method and then the SHA1 algorithm is applied to the output of the decryption. Finally, the CAM can check the CAM client image integrity by comparing the output of the SHA1 algorithm and the delivered hashed value of the CAM client image.

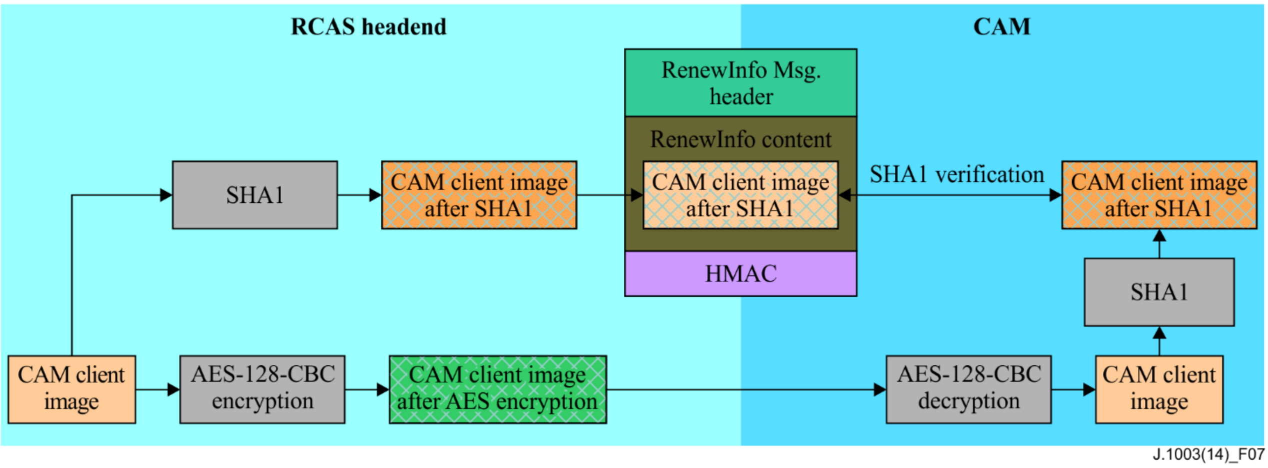


Figure 7 – RCAS client image encryption/decryption and integrity check process

8.3 Security parameters generation

Security parameters used in the RCAS network protocol are generated via the method specified in Table 1.

Table 1 – Security parameter generation method

Parameter	Generation method
NONCE_CAM	Random value with a size of 16 bytes Each session is required to have a unique NONCE_CAM value
RAND_AC	Three random values with a size of 16 bytes Each random value of three of them is required to be unique with respect to each other
IV	Random value with a size of 16 bytes Each session is required to have a unique IV value
Ki	Pre-shared key with a size of 128 bits Each CAM is required to have a unique Ki value
Kc	It has a size of 64 bits and is recommended to be generated using the COMP128 algorithm version 4 (COMP128-4, GSM-MILENAGE))
IHK	IHK = SHA1[RAND_{IHK}||CASS_IP||Key_Paring_ID]
CHK	CHK = SHA1[RAND_{CHK}||CASS_Info]
MEK	See clause 7.3 and Figure 4
ICCIEK	See clause 7.3 and Figure 4
CCCIEK	See clause 7.3 and Figure 5

9 Message

9.1 Message format

It is proposed that the message format between the CASS in the headend and the CAM in the user terminal is specified as shown in the Figure 3. The message format consists of a header, the message contents and the HMAC or message signature for message authentication.

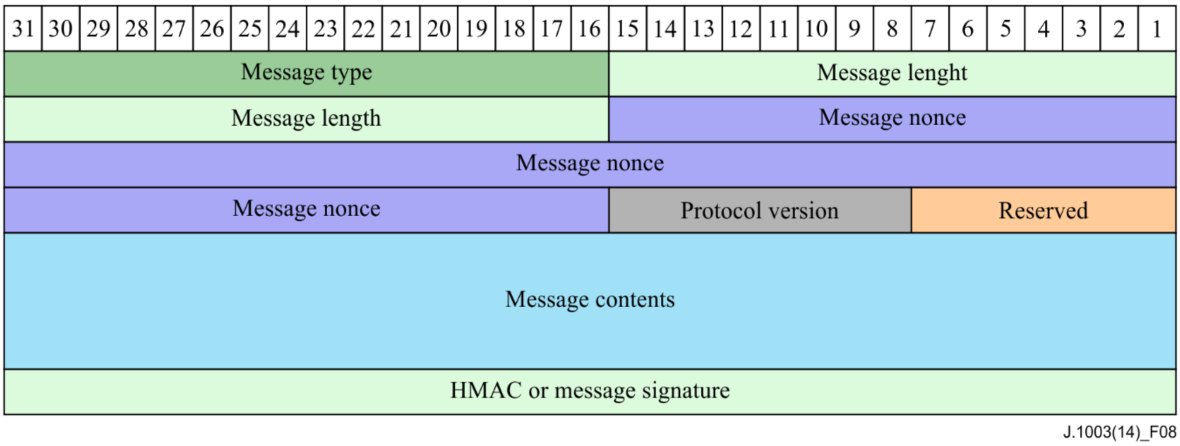


Figure 8 – Proposed message format

It is proposed that the elements in the message header are specified as shown in Table 2. Note that ASN.1 syntax is used for specifying the 'Type (size)'.

Table 2 – Details of the proposed security message header

<table>
<tr><th>Parameters</th><th>Type (size)</th><th>Description</th></tr>
<tr><td>Message type</td><td>OCTET STRING (SIZE (2))</td><td>– 0x0001: RCASAnnounce
– 0x0002: RCASRenewal
– 0x0003: KeyRequest
– 0x0004: KeyResponse
– 0x0005: CAMReg
– 0x0006: CAMRegConfirm
– 0x0007: RenewInfo
– 0x0008: RenewConfirm
– 0x0009: PurchaseReport
– 0x000A: Status
– 0x000B: LogRequest
– 0x000C: LogResponse
– 0x000D: KeyRegRequest
– 0x000E: KeyRegResponse</td></tr>
<tr><td>Message length</td><td>OCTET STRING (SIZE (4))</td><td>The length for both message header and message content. Note that the length of the HMAC and message signature are not included in this parameter.</td></tr>
<tr><td>Message nonce</td><td>OCTET STRING (SIZE (8))</td><td>This is an identifier which is used for preventing reply attack. The message initiator generates the new nonce and delivers it and the copied value of the nonce is used for the reply message. Note that the RCASAnnounce message uses a counter value, which is increased by one whenever a message is changed, as a nonce.
<table>
<tr><td>Security Message</td><td>Usage</td></tr>
<tr><td>RCASAnnounce</td><td>Counter (≠0)</td></tr>
<tr><td>KeyRequest</td><td>Nonce #1</td></tr>
<tr><td>KeyResponse</td><td>Copied nonce #1</td></tr>
<tr><td>CAMReg</td><td>Nonce #2</td></tr>
<tr><td>CAMRegConfirm</td><td>Copied nonce #2</td></tr>
<tr><td>RenewInfo</td><td>Nonce #3</td></tr>
<tr><td>RenewConfirm</td><td>Copied nonce #3</td></tr>
<tr><td>PurchaseReport</td><td>Copied nonce #3</td></tr>
<tr><td>LogRequest</td><td>Nonce #4</td></tr>
<tr><td>LogResponse</td><td>Copied nonce #4</td></tr>
<tr><td>KeyRegRequest</td><td>Nonce #5</td></tr>
<tr><td>KeyRegResponse</td><td>Copied nonce #5</td></tr>
</table></td></tr>
<tr><td>Protocol version</td><td>OCTET STRING (SIZE (1))</td><td>RCAS network protocol version information</td></tr>
<tr><td>Reserved</td><td>OCTET STRING (SIZE (1))</td><td></td></tr>
</table>

9.2 Message fields

The specifications of message fields are described in Annex A and Annex B for normal mode and fast mode respectively.

9.3 Message notations

The message notations are described in Annex A and Annex B for normal mode and fast mode respectively.

Annex A

ASN.1 message notation between CASS and CAM

(This annex forms an integral part of this Recommendation.)

```
-- ============================================================
-- [1] CASS-CAM Security Protocol Message Format (Normal Mode)
-- ============================================================
-- [1-1] Protocol Message Header format : MSB first for all header data transfer
-- [1-1-1] The values of Message Type (2 byte)
-- RCASAnnounce Message ~~~~~~~~~~~0x0001
-- RCASDownload Message~~~~~~~~~~~0x0002
-- KeyRequest Message~~~~~~~~~~~~~0x0003
-- KeyResponse Message~~~~~~~~~~~~0x0004
-- CAMReg Message~~~~~~~~~~~~~~~~~0x0005
-- CAMRegConfirm Message~~~~~~~~~~0x0006
-- RenewInfo Message~~~~~~~~~~~~~~0x0007
-- RenewConfirm Message~~~~~~~~~~~0x0008
-- PurchaseReport Message~~~~~~~~~0x0009
-- Status Message~~~~~~~~~~~~~~~~~0x000A
-- LogRequest Message~~~~~~~~~~~~~0x000B
-- LogResponse Message~~~~~~~~~~~~0x000C

-- [1-2] Protocol Message Content format
-- BER encoding should be used for all messages.
-- But Certificate should be DER encoded.
-- ============================================================
-- ============================================================
CASS-CAM-MESSAGE-FORMAT DEFINITIONS AUTOMATIC TAGS ::= BEGIN
-- ============================================================
-- RCASAnnounceMessage
-- ============================================================
RcasAncMsgContent ::= SEQUENCE
{
-- If the value of protocolTypeFlag is 0x01,
-- the RCAS network protocol works as Normal mode
-- If the value of protocolTypeFlag is 0x02,
-- the RCAS network protocol works as Fast mode
-- The other values are reserved
protocolTypeFlag OCTET STRING (SIZE (1)),
cAMClientVersion SEQUENCE OF CAMClientVersion,
cASSInfo
}
CAMClientVersion ::= SEQUENCE
{
-- CAM chip version
hWVersion OCTET STRING (SIZE (4)),
-- SW version of Bootloader
sWversion OCTET STRING (SIZE (4)),
cAMClientInfo SEQUENCE OF CAMClientInfo
}
CAMClientInfo ::= SEQUENCE
{
-- cAMclientType (CAS, ASD, DRM)
-- 0x01 : CAS (Conditional Access System)
-- 0x02 : ASD (Authorized Service Domain)
-- 0x03 : DRM (Digital Rights Management)
-- 0x04 ~ 0xff : reserved
clientType OCTET STRING (SIZE (1)),
clientPriority OCTET STRING (SIZE (1)),
clientVendor OCTET STRING (SIZE (2)),
clientVersion OCTET STRING (SIZE (2))
}
CASSInfo ::= SEQUENCE
{
cASSUniqueID OCTET STRING (SIZE (4)),
-- Address Type Information
```

```
-- => 0x01: IPv4 Format
-- => 0x02: IPv6 Format
cASSIPAddrType OCTET STRING (SIZE (1)),
-- IP address of CASS
cASSIPAddr OCTET STRING (SIZE (50)),
-- Connection Type Information
-- ==> 0x01: UDP
-- ==> 0x02: TCP
cASSConnectionType OCTET STRING (SIZE (1)),
-- The Listening Port of CASS
cASSListeningPort OCTET STRING (SIZE (4))
}
-- ============================================================
-- RCASRenewalMessage
-- ============================================================
RCASRnMsgContent ::= SEQUENCE
{
-- "YYMMDDhhmmss" - e.g., 2013-May-13, PM 5h50m30s => "130513175030"
downloadSchedule OCTET STRING (SIZE (12)),
-- request = TRUE
keyRequest-Req BOOLEAN (TRUE) OPTIONAL,
purchaseReport-Req BOOLEAN (TRUE) OPTIONAL
}
-- ============================================================
-- KeyRequestMessage
-- ============================================================
KeyReqMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
keyParingID OCTET STRING (SIZE (48)),
-- cAMCertificate MUST be DER encoded
cAMCertificate BIT STRING
}
-- ============================================================
-- KeyResponseMessage
-- ============================================================
KeyRspMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- cASSCertificate MUST be DER encoded
cASSCertificate BIT STRING,
rSAEncryptedContent RSAEncryptedContent,
sIGN-kpk OCTET STRING (SIZE (128))
}
RSAEncryptedContent ::= RSAENCRYPTED{SEQUENCE
{
cHK OCTET STRING (SIZE (20)),
iHK OCTET STRING (SIZE (20)),
rAND-AC-1 OCTET STRING (SIZE (16)),
rAND-AC-2 OCTET STRING (SIZE (16)),
rAND-AC-3 OCTET STRING (SIZE (16)),
kPK OCTET STRING (SIZE (20))
}}
RSAENCRYPTED { ToBeEnciphered } ::= BIT STRING ( CONSTRAINED BY {
-- shall be the result of applying the encipherment procedure --
-- to the BER-encoded octets of a value of -- ToBeEnciphered
} )
-- ============================================================
-- CAMRegMessage
-- ============================================================
CamRegMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
nONCE-CAM OCTET STRING (SIZE (16)),
cAMID OCTET STRING (SIZE (40)),
-- CAM chip version
hWVersion OCTET STRING (SIZE (4)),
-- SW version of Bootloader
sWversion OCTET STRING (SIZE (4)),
hashed-MEK-ICCIEK OCTET STRING (SIZE (20))
}
```

```
-- ============================================================
-- CAMRegConfirmMessage
-- ============================================================
CamRegCnfMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- IVs for 128 AES-CBC
mEK-IV OCTET STRING (SIZE (16)),
iCCIEK-IV OCTET STRING (SIZE (16)),
cCCIEK-IV OCTET STRING (SIZE (16)),
cCCIEK OCTET STRING (SIZE (16))
}
-- ============================================================
-- RenewInfoMessage
-- ============================================================
RenewInfoMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- This field is used for identify CAM Client Image
-- => 0x01 : CAS Client Image
-- => 0x02 : DRM Client Image
-- => 0x03 : ASD Client Image
clientType OCTET STRING (SIZE (1)),
-- This field is used for identify Common IM or Individual IM
-- => 0x01 : Common IM
-- => 0x02 : Individual IM
-- => 0x03 : Deliver Common IM & Individual IM Simultaneously
imageFlag OCTET STRING (SIZE (1)),
-- If the value of imageFlag is 0x01 or 0x02, only one ImageInfo will be generated
-- Otherwise, if the value of imageFlag is 0x03, two ImageInfo will be generated
imageInfo SEQUENCE OF ImageInfo,
-- request = TRUE
purchaseReport-Req BOOLEAN (TRUE) OPTIONAL,
hashed-CCCI OCTET STRING (SIZE (20)),
hashed-ICCI OCTET STRING (SIZE (20)),
-- Control the installation and launching of CAM client image
directives OCTET STRING OPTIONAL
}
ImageInfo ::= SEQUENCE
{
-- Download Server IP address
-- Only when the image is not Common Image
dSIP OCTET STRING (SIZE (16)) OPTIONAL,
-- Transport Mechanism (Carousel, TFTP, HTTP, etc.)
-- 0x01 : Carousel
-- 0x02 : TFTP
-- 0x03 : HTTP
-- 0x04 ~ 0xff : reserved
tm OCTET STRING (SIZE (1)),
-- File Name
-- Only when the image is not Common Image
fn PrintableString OPTIONAL
}
-- ============================================================
-- RenewConfirmMessage
-- ============================================================
RenewCnfMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- success = TRUE, fail = FALSE
downloadStatus BOOLEAN
}
-- ============================================================
-- PurchaseReportMessage
-- ============================================================
PrchsRptMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
purchaseInfo OCTET STRING
}
-- ============================================================
-- StatucAMessage
-- ============================================================
StcAMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- Status Info: 0x01 (CAM-DSC ID validation fail)
-- 0x02 (MEK or ICCIEK hash verification fail)
-- 0x03 (CCI hash verificcation fail)
-- 0x04 (HMAC verification fail)
-- 0x05 (RSA signature verification fail)
-- 0x06 ~ 0xff (reserved)
statusInfo OCTET STRING (SIZE (1))
}
-- ============================================================
-- LogRequestMessage
-- ============================================================
LogRqtMsgContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
-- request = TRUE
logRequest-Req BOOLEAN (TRUE)
}
-- ============================================================
-- LogResponseMessage
-- ============================================================
LogRcAMessageContent ::= SEQUENCE
{
sessionID OCTET STRING (SIZE (10)),
logResponseInfo OCTET STRING OPTIONAL
}
END
```

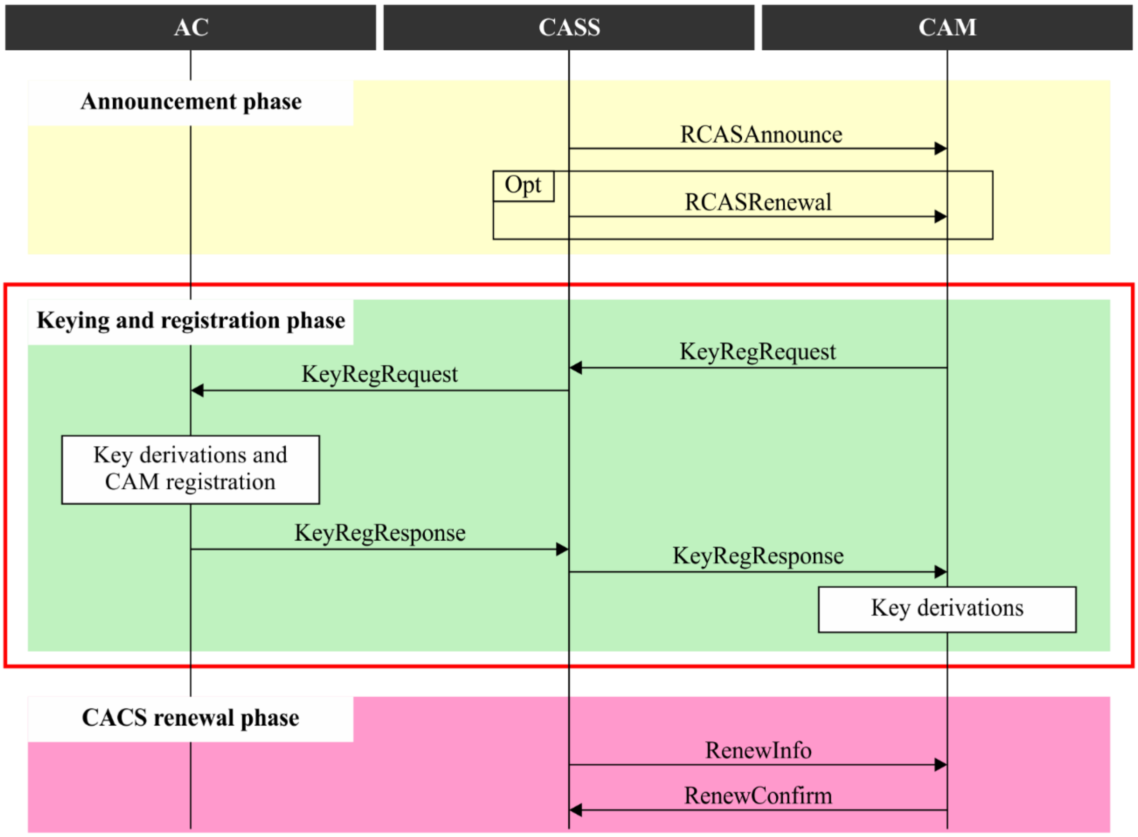
Annex B

Fast mode of network protocol for RCAS

(This annex forms an integral part of this Recommendation.)

Annex B specifies a fast mode of RCAS network protocol. This protocol is a simplified version of normal RCAS network protocol. Service operators who want to use this protocol can select the preferred protocol by using the 'protocolTypeFlag' parameter in 'RcasAncMsgContent'. If the value of the protocolTypeFlag is '0x01', the RCAS network protocol works as normal mode. On the other hand, if the value of the protocolTypeFlag is '0x02', the RCAS network protocol works as fast mode.

B.1 Protocol processing procedures



J.1003(14)_FB.1

B.1.1 KeyRegRequest

Items	Descriptions
Message roles	The RCAS system uses this message in the Keying & Registration phase and the CAM sends it to the CASS and AC. The roles of the message are requesting encryption keys for CACS image and messages as well as the relevant IV values.
Message encryption	None
Message authentication	Digital signature by the CAM
Message processing	1. The CAM sends a KeyRegRequest message that includes the following parameters to the CASS: – Session ID – Key pairing ID – CAM's public key certificate 2. The CASS checks the validity of the received KeyRegRequest message with the following sequences: ① The CASS checks the validity of the CAM's public key certificate through the certificate chains ② If the CAM's public certificate is valid, the CASS authenticates the KeyRegRequest message with a digital signature method ③ If the CASS successfully authenticates the message, it sends the Session ID, Key Pairing ID, CAM ID, CAM HW version, CAM SW version and CAM's public key certificate to the AC. Otherwise, the CASS sends a failure message to the CAM and terminates the RCAS network protocol. 3. The AC registers the CAM ID after receiving the KeyRegRequest message from the CASS and generates RCAS network protocol relevant encryption keys and IVs. Note that the definitions of all keys except the authorization key (AK) are the same as the ones used in the normal RCAS network protocol: – AK – ICCIEK and ICCIEK's IV – CCCIEK and CCCIEK's IV – MEK and MEK's IV – KPK
Key generation method	The key generation methods are as follows: Note that SHA-1(x | y) denotes the result of applying the SHA-1 function to the concatenated bit strings x and y and Truncate(x,n) denotes the result of truncating x to its left-most n bits 1. AK is generated using the method below: Note that AK_PAD is a SHA-1 padding value and it is generated by repeating 0xA3 63 times. Pseudo random number generation function (PRF) is a pseudo random number generation function – AK = Truncate(PRF (SHA-1 (AK_PAD | CAM_ID | Session_ID | NONCE_CAM | HW_Version | SW_Version)),128) 2. ICCIEK is generated using the method below: Note that ICCIEK_PAD is a SHA-1 padding value, and it is generated by repeating 0xA6 63 times. – ICCIEK = Truncate (SHA-1(ICCIEK_PAD | AK), 128) 3. CCCIEK is generated using the method below: Note that CCCIEK_PAD is a SHA-1 padding value and it is generated by repeating 0xAC 63 times.

Items	Descriptions
	– CCCIEK = Truncate (SHA-1(CCCIEK_PAD | AK), 128) 4. MEK is generated using the method below: Note that MEK_PAD is a SHA-1 padding value and it is generated by repeating 0x3A 63 times. – MEK = SHA-1(MEK_PAD | AK), 128) 5. MAK is generated using the method below: Note that MAK_PAD is a SHA-1 padding value and it is generated by repeating 0x6A 63 times. – MAK = SHA-1(MAK_PAD | AK), 128) 6. Key pairing key (KPK) is generated using the method: Note that KPK_PAD is a SHA-1 padding value, and it is generated by repeating 0xCA 63 times. – KPK = SHA-1(KPK_PAD | AK), 128)

ASN.1 syntax
(CAM → CASS)
	`-- =======================================`
	`-- KeyRegRequestMessage`
	`-- =======================================`
	`KeyRegReqMsgContent ::= SEQUENCE`
	`{`
	`sessionID OCTET STRING (SIZE (10)),`
	`keyParingID OCTET STRING (SIZE (48)),`
	`cAMID OCTET STRING (SIZE (40)),`
	`-- CAM chip version`
	`hWVersion OCTET STRING (SIZE (4)),`
	`-- SW version of Bootloader`
	`sWversion OCTET STRING (SIZE (4)),`
	`-- cAMCertificate MUST be DER encoded`
	`cAMCertificate BIT STRING`
	`}`

B.1.2 KeyRegResponse

Items	Descriptions
Message roles	The RCAS system uses this message in the Keying & Registration phase and the CASS sends it to deliver AK to the CAM.
Message encryption	Only AK is encrypted using the CAM's public key
Message authentication	digital signature per CASS
Message processing	1. The AC generates the listed parameter below and sends the Session ID and AK to the CASS – Session ID – AK – ICCIEK – ICCIEK IV – CCCIEK – CCCIEK IV – MEK – MEK IV – MAK – MAK IV

Items	Descriptions
	– KPK 2. The CAM performs the message authentication process using a digital signature method 3. If the CAM can successfully verify the message authentication, it decrypts AK from the message using the CAM's private key 4. The CAM generates the listed parameters below using the same key generation methods as the ones used in the AC. – ICCIEK – ICCIEK IV – CCCIEK – CCCIEK IV – MEK – MEK IV – MAK – MAK IV – KPK

ASN.1 syntax
(CASS → CAM)
	`-- =======================================`
	`-- KeyRegResponseMessage`
	`-- =======================================`
	`KeyRegRspMsgContent ::= SEQUENCE`
	`{`
	`sessionID OCTET STRING (SIZE (10)),`
	`rSAEncryptedContent`
	`}`
	`RSAEncryptedContent ::= RSAENCRYPTED{SEQUENCE`
	`{`
	`aK OCTET STRING (SIZE (16))`
	`}}`
	
	`RSAENCRYPTED { ToBeEnciphered } ::= BIT STRING ( CONSTRAINED BY {`
	`-- shall be the result of applying the encipherment procedure -- to the BER-encoded octets of a value of -- ToBeEnciphered`
	`} )`

Annex C

CAM cloning detection process

(This annex forms an integral part of this Recommendation.)

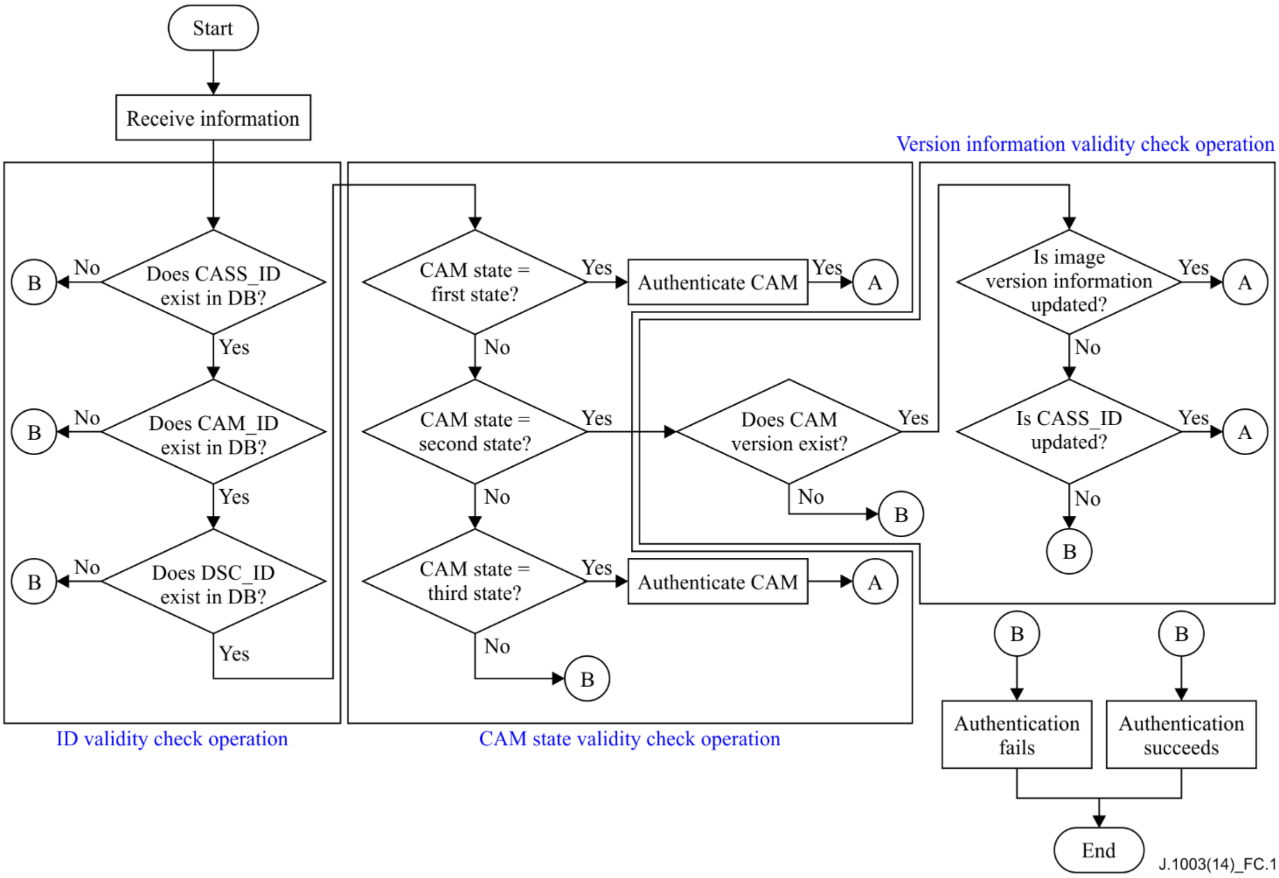


Figure C.1 – CAM cloning detection process in an AC when a 'RenewConfirm' is not received from a CASS

Figure C.1 is a flowchart illustrating a CAM copy detection method using an AC. The CAM copy detection method of Figure C.1 is performed when a 'RenewConfirm' is not received from a CASS. In other words, when the CASS has a function of reusing a CAM authentication result received from the AC, instead of deleting the CAM authentication result, until a CAM receives the 'RenewConfirm', the CAM copy detection method of Figure C.1 is performed. The CAM copy detection method of Figure C.1 broadly includes an 'ID validity check operation', a 'CAM state validity check operation' and a 'version information validity check operation'.

In the 'Receive information' operation, the AC receives, from at least one CASS, the CAM_ID, CASS_ID and DSC_ID.

In the 'ID validity check operation', the AC determines whether a CASS_ID, a CAM_ID and a DSC_ID that are received from the at least one CASS exist in a DB. When the CASS_ID, the CAM_ID and the DSC_ID are determined not to exist in the DB, the AC determines the value of the 'Auth_Rst' to be 'failure'.

In the 'CAM state validity check operation', the AC performs a second check operation of checking the validity of the CAM based on state information. Here, the state information of a CAM is received from the at least one CASS. Additionally, the state information of the CAM includes first state information indicating a state before a RCAS service is provided to a CAM, second state information indicating a state where a CAM is included in at least one MSO and joins the RCAS service and third state information indicating a state where a CAM is withdrawn from the RCAS service. Specifically, when an authentication request is received from a CAM in a 'Virgin' state, the AC changes the 'Virgin' state of the CAM to an 'Auth_Service' and sets the value of the 'Auth_Rst' to be 'success'. When an authentication request is received from a CAM in the 'Auth_Service' state, the AC performs a third check operation. Furthermore, when an authentication request is received from an CAM in an 'Auth_Not_Service' state, the AC changes the 'Auth_Not_Service' state of the CAM to the 'Auth_Service' state and sets the value of the 'Auth_Rst' to be 'success'.

In the 'version information validity check operation', the AC performs the third check operation of checking the validity of a CAM based on the version information. Here, the AC performs the third check operation, only when a CAM requesting authentication is in the 'Auth_Service' state. First, the AC downloads in the DB, hardware and software version information corresponding to an ID value of the CAM that requests authentication. Subsequently, the AC determines whether the hardware and software version information called from the DB is identical to the hardware and software version information for the CAM received from the at least one CASS. When the called hardware and software version information is determined to be identical to the received hardware and software version information, the AC proceeds to a next operation. Conversely, when the called hardware and software version information is determined to differ from the received hardware and software version information, the AC sets the value of the 'Auth_Rst' to be 'failure'. Additionally, the AC determines whether CAS image version information is updated, only when the received hardware and software version information is determined to be identical to the hardware and software version information stored in the DB. Here, the updated CAS image version information indicates that the CAS image version information stored in advance in the DB differs from version information newly received from the CASS. When determining that the CAS image version information is not updated, the AC determines whether a CASS-ID value is changed. Specifically, the AC determines whether a CASS-ID received from the CASS differs from a CASS-ID that is stored in the DB and is used to identify a CASS including a corresponding CAM. The AC determines whether the CASS-ID value is changed, to permit authentication of a normal CAM when the normal CAM moves on a CASS zone, despite the CAS image version information being updated. Finally, the AC authenticates the at least one CAM, based on a result of at least one of the first check operation, the second check operation and the third check operation. Here, when the state of the at least one CAM corresponds to the first state information and the third state information in the second check operation, the AC authenticates the at least one CAM. Additionally, when a validity check result value for the 'Version Info' is updated and when the state of the at least one CAM corresponds to the second state information, the AC authenticates a CAM.

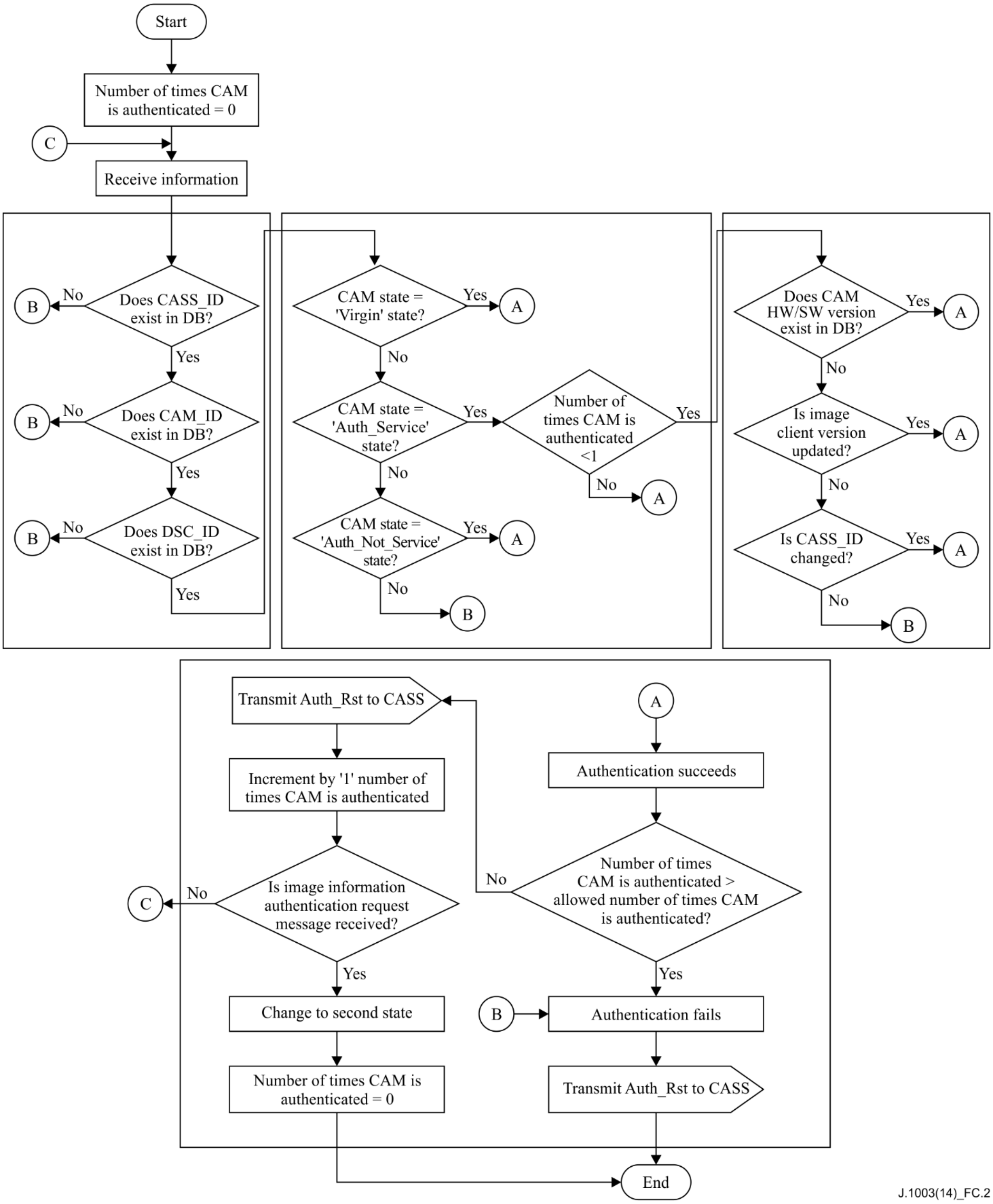

Start
Number of times CAM is authenticated = 0
C
Receive information
Does CASS_ID exist in DB?
No
B
Yes
Does CAM_ID exist in DB?
No
B
Yes
Does DSC_ID exist in DB?
No
B
Yes
CAM state = 'Virgin' state?
Yes
A
No
CAM state = 'Auth_Service' state?
Yes
Number of times CAM is authenticated <1
Yes
No
A
No
CAM state = 'Auth_Not_Service' state?
Yes
A
No
B
Does CAM HW/SW version exist in DB?
Yes
A
No
Is image client version updated?
Yes
A
No
Is CASS_ID changed?
Yes
A
No
B
Transmit Auth_Rst to CASS
Increment by '1' number of times CAM is authenticated
Is image information authentication request message received?
No
C
Yes
Change to second state
Number of times CAM is authenticated = 0
A
Authentication succeeds
Number of times CAM is authenticated > allowed number of times CAM is authenticated?
No
Yes
B
Authentication fails
Transmit Auth_Rst to CASS
End
J.1003(14)_FC.2

Figure C.2 – CAM cloning detection process in AC when a 'RenewConfirm' is received from a CASS

Figure C.2 is a flowchart illustrating a CAM copy detection method using an AC. The CAM copy detection method of Figure C.2 is performed when a 'RenewConfirm' is received from a CASS. In other words, the CAM copy detection method of Figure C.2 is performed when the CASS does not have a function of reusing a CAM authentication result received from the AC instead of deleting the CAM authentication result, until the CAM receives the 'RenewConfirm' message.

In the first operation, the AC checks a number of times that a CAM is authenticated to zero.

In the 'Receive information' operation, the AC receives, from at least one CASS, CAM_ID, CASS_ID, and DSC_ID.

The first check operation of Figure C.2 is performed in the same manner as the first check operation of Figure C.1.

The second check operation of Figure C.2 is performed in the same manner as the second check operation of Figure C.1, except for checking of the number of times that a CAM is authenticated. When the number of times the CAM is authenticated is less than '1', the AC performs a third check operation. Otherwise, the AC performs a fourth check operation of repeatedly checking whether CAM authentication is permitted. Here, a third check operation is performed in the same manner as the third check operation of Figure C.2.

The fourth check operation of Figure C.2 is performed so that the AC repeatedly permit an authentication request from an identical CAM within the number of times the identical CAM is authenticated, until the 'RenewConfirm' is received from the CASS. Here, the number of times is determined in advance by an operator.

Bibliography

[b-ITU-T J.93]	Recommendation ITU-T J.93 (1998), *Requirements for conditional access in the secondary distribution of digital television on cable television systems*.
[b-ITU-T J.193]	Recommendation ITU-T J.193 (2004), *Requirements for the next generation of set-top-boxes*.
[b-ITU-T J.290]	Recommendation ITU-T J.290 (2006), *Next generation set-top box core architecture*.

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	**Cable networks and transmission of television, sound programme and other multimedia signals**
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2015