INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 2013-2016

STUDY GROUP 17

TD 1634 Rev.1

English only

Original: English

Question(s):	PLEN/17, 11/17, (7/17)	Geneva, 8 - 17 April 2015

TD

Source:	Editors
Title:	Draft new Recommendation ITU-T X.1341 (X.cmail) (Last Call Comment Resolution; for consideration)

Draft Recommendation ITU-T X.1341 (X.cmail) was sent to AAP Last Call #45 on 1 November 2014. Comments from France, Germany, and Korea (Republic of) were received by the deadline of 28 November 2014 against draft Rec. ITU-T X.1341 (X.cmail), and the text was put into Last Call Judgement to allow resolution of the comments.

However, not all comments could be resolved during Last Call Judgement, and hence, draft Rec. ITU-T X.1341 (X.cmail) was not approved as announced in AAP #49 on 16 January 2015 and the material was forwarded to the April 2015 Study Group 17 meeting for further consideration.

This TD holds

- In the Annex: Revised draft Recommendation ITU-T X.1341 (X.cmail) with changes made due to resolved Last Call comments.
- In attachment 1: The AAP Last Call comments and the status of their resolution.

Note – *Germany believes that draft new Recommendation ITU-T X.1341 (X.cmail) has, beyond doubt, policy or regulatory implications; and Germany proposed that the Alternative Approval Process is not to be applied and "the approval procedure shall proceed according to Resolution 1 clause 9.3 …" (see Rec. ITU-T A.8, section 5.2).*

Attachment: 1

- Comment resolution sheet.

Contact	Laura Prin LegalBox SA France	Tél: +33 1 43 20 39 38 E-mail: laura.prin@legalbox.com

Annex

Draft new Recommendation ITU-T X.1341 (X.cmail)

Certified mail transport and certified post office protocols

Summary

The objective of this Recommendation ~~ITU-T X.1341~~ is to define~~of~~ the certified mail transfer protocol (CMTP) and certified post office protocol (CPOP) in order~~is~~ to foster the exchanges of electronic certified mails in the world in a secure way by providing confidentiality, identification of the correspondents, integrity and non-repudiation.

Table of Contents

Introduction

This Recommendation extends the capabilities of simple mail transfer protocol (SMTP) and post office protocol version 3 (POP3) to support authentication, security and non-repudiation~~ to make e-mails legally binding~~.

For this purpose, two protocols are specified:

– the certified mail transfer protocol (CMTP), which is an extension to the simple mail transfer protocol (SMTP), is the protocol supporting the communications between the sender of e-mails and a mail server, called the certified mail (Cmail) server;

– the certified post office protocol (CPOP), which is an extension to the post office protocol version 3 (POP3), is the protocol supporting the communications between the recipient of e-mails and the Cmail server.

Within SMTP and POP3 a message type is identified by a commands, i.e., a key word at the start of the message. For CMTP and CPOP, new commands have been defined and some of the SMTP and POP3 commands has been extended. In particular, some commands have been extended to carry notices (electronic documents) allowing to document and verify the different stages of the communication from the sender to the recipient.

CMTP and CPOP also introduce the concept of Cmail server that is an active partner in the communication between the sender and the recipient allowing it to certify that the exchange between two parties has indeed occurred.

Certified mail assumes that an existing public-key infrastructure (PKI) is established.

Annex A, which is an integral part of this Recommendation, provides the formal specification for notices using the XML schema definition (XSD) notation technique.

Annex B, which is an integral part of this Recommendation, provides the formal specification for notices using the abstract syntax notation one (ASN.1).

Annex C, which is an integral part of this Recommendation, specifies the requirements on public-key certificates issued to clients (sender and recipient of e-mails) and Cmail servers.

Annex D, which is an integral part of this Recommendation, specifies requirements on the use of the transport layer security (TLS) specification.

Annex E, which is an integral part of this Recommendation, specifies object identifiers defined for Cmail server.

1 Scope

The scope of this Recommendation is to provide a specification on how to make e-mails ~~legally~~ trusted binding.

Certified mail transfer protocol/certified post office protocol (CMTP/CPOP) enables to:

– solve repudiation issues because of the use of electronic signature;

– solve confidential issues because of the use of encryption;

– produce reliable notices of deposit, notices of transit and notices of reception;

– use a certified mail (Cmail) server to track certified mails to avoid their loss during the process;

– use a transport layer security (TLS) connection to provide stronger identification. This stronger level of identification is required by the Cmail server.

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 X.520]	Recommendation ITU-T X.520 (2012) | ISO/IEC 9594-6:2014, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*.
[ITU-T X.680]	Recommendation ITU-T X.680 (2008) | ISO/IEC 8824-1:2008, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation*.
[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).*
[ITU-T X.693]	Recommendation ITU-T X.693 (2008) | ISO/IEC 8825-4:2008, *Information technology – ASN.1 encoding rules: XML Encoding Rules (XER).*
[ISO 3166-1]	ISO 3166-1:2013, *Codes for the representation of names of countries and their subdivisions – Part 1: Country codes.*
[IETF RFC 822]	IETF RFC 822 (1982), *Standard for the Format of ARPA Internet Text Messages*.
[IETF RFC 1939]	IETF RFC 1939 (1996), *Post Office Protocol – Version 3.*
[IETF RFC 2045]	IETF RFC 2045 (1996), *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies*.
[IETF RFC 5246]	IETF RFC 5246 (2008), *The Transport Layer Security – Protocol Version 1.2.*
[IETF RFC 5321]	IETF RFC 5321 (2008), *Simple Mail Transfer Protocol.*
[XML]	W3C Recommendation XML1.0 (2000), *Extensible Markup Language (XML) 1.0 (Second Edition)*.
[XSD]	W3C Recommendation XML Schema (2001), *XML Schema Part 1: Structures*.

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 certification authority (CA) [b-ITU-T X.509]: An authority trusted by one or more users to create and assign public-key certificates. Optionally the certification authority may create the subjects' keys.

3.1.2 certificate validation [b-ITU-T X.509]: The process of ensuring that a certificate was valid at a given time, including possibly the construction and processing of a certification path, and ensuring that all certificates in that path were valid (i.e., were not expired or revoked) at that given time.

3.1.3 hash function [b-ITU-T X.509]: A (mathematical) function which maps values from a large (possibly very large) domain into a smaller range. A "good" hash function is such that the results of applying the function to a (large) set of values in the domain will be evenly distributed (and apparently at random) over the range.

3.1.4 post office protocol v3 (POP3) [IETF RFC 1939]: ~~TCP/IP~~ A~~a~~pplication layer protocol over the TCP/IP connection ~~protocol~~ used to receive e-mail.

3.1.5 private key [b-ITU-T X.509]: (In a public key cryptosystem) that key of an entity's key pair which is known only by that entity.

3.1.6 public key [b-ITU-T X.509]: (In a public key cryptosystem) that key of a user's key pair which is publicly known.

3.1.7 public-key certificate (PKC) [b-ITU-T X.509]: The public key of a user, together with some other information, rendered unforgeable by digital signature with the private key of the CA which issued it.

3.1.8 public-key infrastructure (PKI) [b-ITU-T X.509]: The infrastructure able to support the management of public keys able to support authentication, encryption, integrity or non-repudiation services.

3.1.9 simple mail transfer protocol (SMTP) [IETF RFC 5321]: A~~a~~pplication layer protocol over the TCP/IP connection ~~TCP/IP protocol~~ used to send e-mail.

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 certified mail: Electronic mail exchanged using certified mail transfer protocol (CMTP) and certified post office protocol (CPOP).

3.2.2 certified mail transfer protocol (CMTP): A~~a~~pplication layer protocol over the TCP/IP connection ~~Transmission control protocol/Internet protocol (TCP/IP)~~ based on SMTP used to send certified mail.

3.2.3 certified post office protocol (CPOP): A~~a~~pplication layer protocol over the TCP/IP connection based on POP3 ~~Transmission control protocol/Internet protocol (TCP/IP)~~ used to receive certified mail.

3.2.4 cmail server: Trusted ~~Legal~~ entity involved in certified mail transactions.

3.2.5 notice of deposit: Electronic document signed by the sender and the Cmail server, containing information allowing to certify that a certified mail deposit occurred.

3.2.6 notice of reception: Electronic document signed by the recipient and the Cmail server, containing information allowing to certify that a certified mail was received by the recipient.

3.2.7 notice of transit: Electronic document signed by the Cmail servers involved in the transaction and containing information allowing to certify that the certified mail was transmitted to the Cmail server.

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AES	Advanced Encryption Standard

ASN.1	Abstract Syntax Notation One
CA	Certification Authority
CBC	Cipher Block Chaining
CK	Cypher Key
Cmail	Certified Mail
CMTP	Certified Mail Transfer Protocol
CPOP	Certified Post Office Protocol
DER	Distinguished Encoding Rules
DNS	Domain Name System
ECK	Encrypted Cipher Key
ESMTP	Extended Simple Mail Transfer Protocol
id	Identity
IP	Internet Protocol
MIME	Multipurpose Internet Mail Extensions
PKI	Public-Key Infrastructure
POP3	Post Office Protocol version 3
RSA	Rivest, Shamir and Adleman algorithm
RSCK	Random Symmetric Cypher Key
SHA	Secure Hash Algorithm
S/MIME	Secure/Multipurpose Internet Mail Extensions
SMTP	Simple Mail Transfer Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
UCS	Universal Character Set
UTF-8	UCS Transformation Format-8
XER	XML Encoding Rules
XML	eXtensible Markup Language
XSD	XML Schema Definition

5 Conventions

None.

6 Certified mail basic concepts

In traditional e-mail communications using the simple mail transfer protocol (SMTP) and post office protocol v3 (POP3) a recipient of an e-mail can deny ever having received it. This is even the case when secure/multipurpose internet mail Extensions (S/MIME) is added to the protocol suite.

S/MIME provides for encryption of messages and authentication of the sender, but it does not provide proof of delivery.

This Recommendation is specification for a protocol suite called certified mail and is comprised of the certified mail transfer protocol (CMTP) and the certified post office protocol (CPOP).

In a SMTP/POP3 communications, the mail server is not an active part in the communications, but is only forwarding messages as they are received when the recipient signs on to the mail server. This is even the case when SMIME is employed.

In certified mail, the mail server is actively participating in the communication between the sender and the recipient in a way that allows the Cmail server to verify that the recipient has accepted to receive the mail. The mail is sent encrypted not allowing the Cmail server to read the actual content of the e-mail. An overview of procedure is given in the following, while a detailed specification is given in clause 8.

The interactions between the sender and the Cmail server is specified in clause 8.

The interactions between the recipient and the Cmail server is specified in clause 9.

7 Types of certified mail commands

Certified mail makes use of a combination of current SMTP and POP3 commands, some enhanced SMTP and POP3 commands and some certified mail specific command. In tables 1 and 2, commands that do not have the counterpart in SMTP/POP3 are labeled "Additional". Commands that are enhanced SMTP/POP3 commands are labeled "Modified". SMTP/POP3 commands that are used unchanged are labeled "Unchanged".

A command type is defined as a keyword using upper case letters that identifies a particular message type together with some additional specifications for that message type.

7.1 Types of CMTP commands

Table 1 – CMTP commands

Command	Command function
CELO Additional	Enables the server to identify its processing of CMTP commands.
DELV Additional	Identifies the delivery mode: certifiedMail.
MAIL FROM Modified	Identifies the sender of the message; used as “MAIL FROM”. If the account exists on the server, then it sends back a base64 of the public-key certificate of the known sender.
RCPT TO Modified	Identifies the recipients of the message; used under “RCPT TO” format. If the account exists on the server, then it sends back a base64 of the public-key certificate of the known sender. If the account exists on another CMTP server with which key exchanges have been made, then the server questions the second server and sends a base64 of the public-key certificate belonging to the known recipient with the CHCK RCPT command.
CHCK RCPT Additional	Sent only if the recipient is attached to another Cmail server than the Cmail server for the sender.

DATA Modified	Sent by a client to initiate the transfer of message content. The server sends back in a notice of deposit signed by the server and to be signed by the sender.
DEPO Additional	Sent by a client to initiate the transfer of the notice of deposit content signed by the server and countersigned by the sender.
SEND EVLP Additional	Forwards envelope from one Cmail server to another.
HELP Unchanged	Returns a list of commands that are supported by the CMTP server.
QUIT Unchanged	Terminates the session.

7.2 Types of CPOP commands

Table 2 – CPOP commands

Command	Command function
USER Unchanged	Used to specify the name of the user who is logging on.
PASS Unchanged	Password of the user who is logging on.
LIST Modified	Used to list messages and their combined size. For example, invoking the LIST command with no parameters will return 2 +OK messages (320 octets), and the list of messages: identity (id), length and delivery mode (if any) like CertifiedMail.
RETR Modified	Where N is a number between 1 and the last number returned by the LIST command. This command may not be used to retrieve a message that has been marked as deleted. If there is no delivery type, the server sends the e-mail in multipurpose Internet mail extensions (MIME) encoding. If delivery mode is defined, the server processes the message specifically. For example with CertifiedMail, the server challenges the recipient before sending the envelope by using RCPT command.
CHLG RESP Additional	Sent by the client to give notice of reception for the message and give the reply to the secret question. If the reply is correct then the server sends back the MIME envelope.
SEND NORP Additional	Sends the signed notice of reception.

Command	Command function
HELP Unchanged	Returns a list of command that is supported by the CPOP server.
QUIT Unchanged	Terminates the session.

8 Detailed CMTP specification

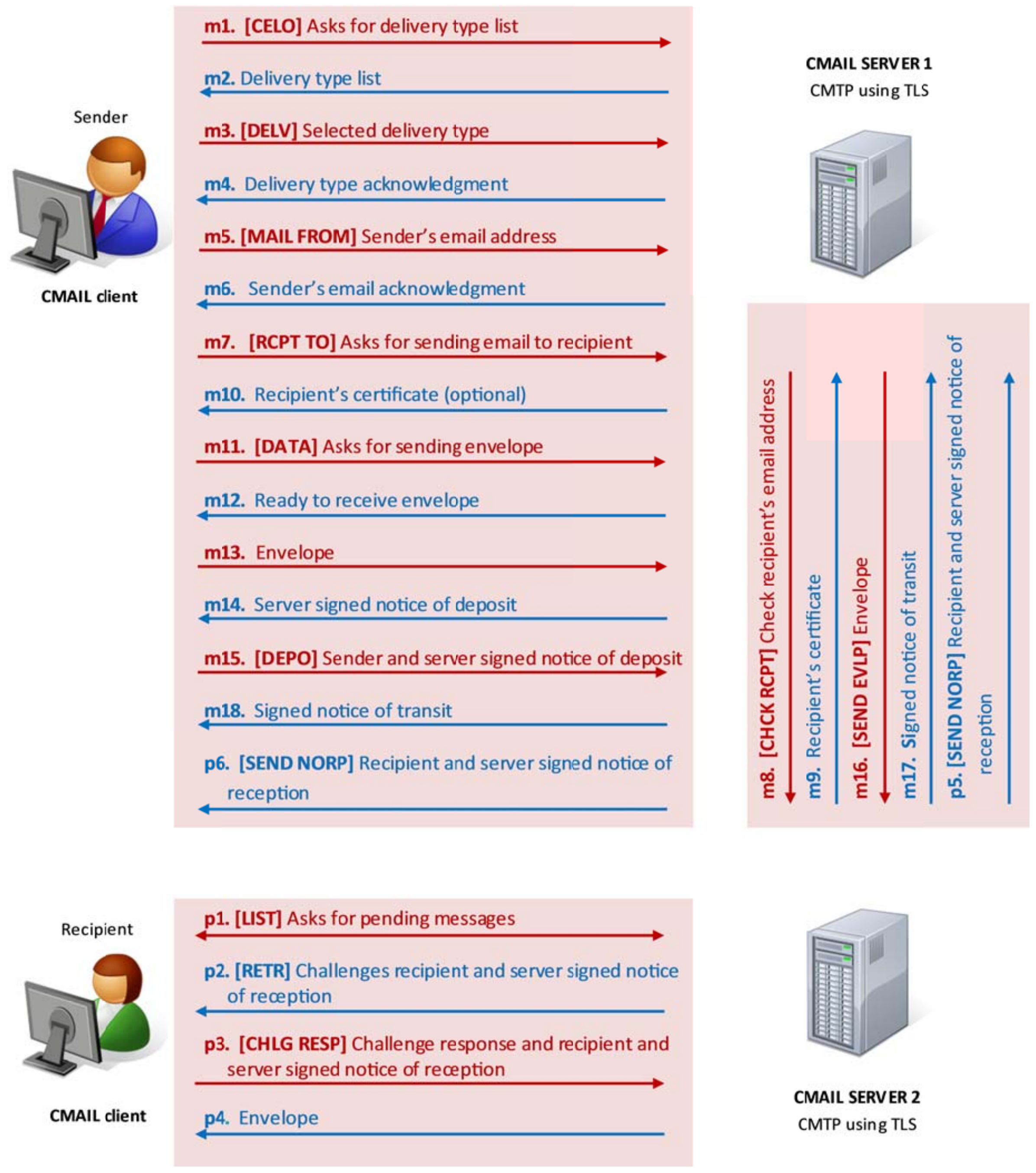

m1. [CELO] Asks for delivery type list
m2. Delivery type list
m3. [DELV] Selected delivery type
m4. Delivery type acknowledgment
m5. [MAIL FROM] Sender's email address
m6. Sender's email acknowledgment
m7. [RCPT TO] Asks for sending email to recipient
m10. Recipient's certificate (optional)
m11. [DATA] Asks for sending envelope
m12. Ready to receive envelope
m13. Envelope
m14. Server signed notice of deposit
m15. [DEPO] Sender and server signed notice of deposit
m18. Signed notice of transit
p6. [SEND NORP] Recipient and server signed notice of reception
Sender
CMAIL client
CMAIL SERVER 1
CMTP using TLS
m8. [CHCK RCPT] Check recipient's email address
m9. Recipient's certificate
m16. [SEND EVLP] Envelope
m17. Signed notice of transit
p5. [SEND NORP] Recipient and server signed notice of reception
Recipient
CMAIL client
p1. [LIST] Asks for pending messages
p2. [RETR] Challenges recipient and server signed notice of reception
p3. [CHLG RESP] Challenge response and recipient and server signed notice of reception
p4. Envelope
CMAIL SERVER 2
CMTP using TLS

Figure 1 – Overview of protocol exchanges

Commands prefixed by "m" are used in the CMTP protocol, and commands prefixed by "p" are used in the CPOP protocol. Clause 8.1 gives a detailed specification for exchanges m1 to m18 in Figure 1, while clause 8.2 gives a detailed specification for exchanges p1 to p6.

8.1 CELO: Ask for delivery type list

The command type is sent as a SMTP message, similar to the HELO command, followed by a fully qualified domain name. Its purpose is to retrieve a list of delivery types.

8.2 Delivery type list

The delivery type list is given in response to the CELO command. It is in SMTP format with the following content (case insensitive):

250-<Fully qualified domain name of the Cmail server>

250-8BITMIME

250-Delivery-Types CertifiedMail <other delivery types>

250 OK

This Recommendation only makes specification for CertifiedMail. Future editions may specify other delivery types.

8.3 Selected delivery type

This message identifies the delivery type of the ones specified in the delivery type list. It has the following (SMTP) format:
DELV <delivery type>

8.4 Delivery type acknowledgement

In the case the selected delivery type is accepted, this message has the following SMTP format (case insensitive):

250 Delivery-Type <delivery type>OK

The following response is given in case of a syntax error in the selected delivery message:

501 Syntax: DELV <delivery type>

The following response is given when the selected delivery message was issued out of sequence:

501 Syntax: use CELO command first

The following response is given when the selected delivery message was unknown:

501 Unknown Delivery-Type: <delivery type>

8.5 Sender's e-mail address

This message is sent to the Cmail server to request sending a certified mail and optionally to request the sender's public-key certificate from the Cmail server.

MAIL FROM <sender's email address> [CertificateRequested]

8.6 Sender's e-mail acknowledgement

This message is sent to confirm that the sender's e-mail address exists in the Cmail server database. If the sender requested its public-key certificate, the sender's public-key certificate is included:

[250 User-Certificate: <public-key certificate encoded in Base64>]

250 OK

8.7 Ask for sending e-mail to recipient

This message is sent to the Cmail server to request sending one certified mail to the recipient and optionally to request the recipient's public-key certificate from the Cmail server.

RCPT TO <recipient's email address> [CertificateRequested]

This command may be used as many times as necessary in order to add each recipient if there are several recipients. The information indicating whether the recipient is “To” or “Cc” is contained in the header of the envelope [IETF RFC 5321]. “Cci” recipients are not allowed.

8.8 Check recipient's e-mail address by the remote Cmail server

This message is only sent if the recipient is attached to another Cmail server other than the Cmail server for the sender. It is sent from the sender's Cmail server to the recipient's Cmail server to check the validity of the e-mail address and optionally to request the recipient's public-key certificate.

CHCK RCPT <recipient's email address> [CertificateRequested]

8.9 Recipient's e-mail address acknowledgement

This message is sent in response to "Check recipient's e-mail address by the remote Cmail server".

The following confirms the e-mail address and includes the recipient's public-key certificate if so requested:

[250 User-Certificate: <public-key certificate encoded in Base64>]

250 OK

In case the e-mail address cannot be confirmed, the following error message may be sent:

503 Sender already specified

shall be sent if it is a response to a duplicate request.

501 Syntax: CHCK RCPT <address>

shall be sent if there is a syntax error in the recipient's e-mail address.

501 Syntax: CHCK RCPT <address> Error in parameters <parameter>

shall be sent if the parameter after the e-mail address was not recognized.

553 <email address> Invalid email address

shall be sent if the e-mail address does not exist at the remote Cmail server.

8.10 Recipient's e-mail acknowledgement

This message is sent to confirm that the recipient's e-mail address exists. If the sender requests the recipient's public-key certificate, the recipient’s public-key certificate is included.

The following confirms the e-mail address and includes the recipient's public-key certificate if so requested:

[250 User-Certificate: <public-key certificate encoded in Base64>]

250 OK

In case the e-mail address cannot be confirmed, the following error message may be sent:

503 Error: need MAIL FROM command

shall be sent if the message was sent out of sequence.

452 Error: too many recipients

shall be sent if too many recipients were specified.

501-6.1.1 Syntax: RCPT TO <address>

shall be sent if there is a syntax error in the recipient's e-mail address.

501-6.1.2 Syntax: RCPT TO <address> Error in parameters: <parameters>

shall be sent if the parameter after the e-mail address was not recognized.

550-5.1.1 <email address> Invalid email address.

shall be sent if the e-mail address does not exist.

8.11 Ask for sending ENVELOPE

The following format is used by the sender to ask the Cmail server permission to send data

DATA

8.12 Ready to receive ENVELOPE

The following message is sent if the Cmail server is ready to receive data:

354 Start mail input; end with <CRLF>.<CRLF>

The following message is sent when the MAIL FROM command has not been sent:

503 Error: need MAIL FROM command

The following message is sent when the RCPT TO command has not been sent:

503 Error: need RCPT TO command

The following message is sent when the DELV command has not been sent:

503 Error: need DELV command

8.13 ENVELOPE

The client shall:

1. generate a random symmetric cipher key (RSCK), e.g. advanced encryption standard (AES) 256;
2. encrypt the body of the message and attachments, if any, using this key;
3. build a MIME message containing a part named ENVELOPE which contains the encrypted message (see [IETF RFC 2045]);
4. end the message with <CR><LF>.<CR><LF>; and
5. send the MIME message.

8.14 Server signed notice of deposit

250 Notice-of-deposit:

<notice of deposit signed by the Cmail server encoded in base64>

250 Ok

The server generates a notice of deposit containing information about the envelope (envelope id, delivery type and mime hash), and signs it with its private key.

8.15 Sender and server signed notice of deposit

The sender shall:

1. decode the received notice of deposit;
2. build challenge for each recipients;

3. sign the server signed notice of deposit using its own private key;
4. encode the result in base64; and
5. transmit it to the Cmail server using:

   DEPO <notice of deposit base64 encoded>

The challenge is defined in Annex A, figure A.6 – challenge.

The challenge contains the **`SecretQuestion`**, **`CipherEnvelopeKey`**, and the public-key certificate of the recipient.

`SecretQuestion`: is composed by a **`Request`** and a **`Response`**.

The Request may contain a **`RandomNumber`**. The Response contains the **`AlgorithmIdentifier`** to be recalculated by the sender in order to receive the ENVELOPE. This **`AlgorithmIdentifier`** identifies the algorithm used to compute the hash. The challenge consists of first recovering the cipher key RSCK, ciphered by the public key of the recipient, then concatenating the **`RandomNumber`** and RSCK, and computing the hash to build the response.

Example of a challenge in extensible markup language (XML):

```
<Entity EmailAddress="john.doe@example.org" Type="to">
  <SecretQuestion>
    <Request RandomNumber="30987497498789739837"/>
   <Response AlgorithmIdentifier="2.16.840.1.101.3.4.2.1"
Encoding="base64">5mYZWhtl0yxBa/wl7VLiiQ=</response>
  </SecretQuestion>
  <CipherEnvelopeKey Algorithm="AES" CipheredKey="RSA" Encoding="base64-DER"
KeySize="256">UjBg…b1PHDOOM4IFnTpzHn9TQ==</cipherEnvelopeKey>
  <Certificate
Encoding="base64">MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM…sdjn7VDBlb+WS10j2rJcAHHsUyr…
/gy7</Certificate>
</Entity>
```

NOTE 1 – This challenge could use abstract syntax notation one (ASN.1) distinguished encoding rules (DER).

NOTE 2 – The server is not able to recalculate the hash since it does not know the encryption key. However, only the server knows the expected result from the hash calculation.

NOTE 3 – During the challenge with the recipient, the server sends only the secret question and waits for the recipient's reply.

8.16 ENVELOPE between Cmail servers

The message defined in clause 8.13 is forwarded to another Cmail server only if the sender and the recipient are attached to different Cmail servers (see item m16 in Figure 1).

SEND EVLP <MIME message>

8.17 Signed notice of transit between Cmail servers

The following format shall be used:

250 Notice-of-transit:

<notice of transit base64 encoded>

The following message is sent if the Cmail server receives a notice of transit:

250 Ok

The following message is sent when the notice of transit is incorrect:

503 Error: incorrect Notice-of-transit

Notice of transit is built by the Cmail that received the ENVELOPE.

This Cmail server generates a notice of deposit containing information about the envelope (envelope id, delivery type and mime hash), and signs it with its private key. This notice is the same as the notice of deposit.

8.18 Signed notice of transit

The Cmail sender server shall:

1. decode the received notice of transit;
2 sign the server signed notice of transit using its own private key;
3. encode the result in base64; and
4. transmit it to the Cmail server using:

250 Signed-notice-of-transit:

<signed notice of transit base64 encoded>

250 Signed-notice-of-deposit:

<signed notice of deposit base64 encoded>

250 Ok

9 Certified post office protocol (CPOP)

Below is an explanation for p1 to p6 of Figure 1.

9.1 Ask for pending messages

Information on pending messages is performed using the procedure specified in section 5 under LIST command in [IETF RFC 1939] with an additional parameter. For each line detailing a pending message, the additional parameter is added indicating the delivery type if it is not a standard e-mail (see item p1 in Figure 1). Example:

C: LIST

S: +OK 2 messages (320 octets)

S: 1 120

S: 2 200 CertifiedMail

S: .

This procedure also includes retrieving all standard e-mails leaving only messages tagged with delivery type on the Cmail server.

9.2 Challenge recipient and server signed notice of reception

For messages tagged with delivery type, the RETR command does not retrieve the message but retrieves the challenge and the server signed notice of reception base64 encoded. The client verifies the digital signature and the sender certificate contained in the notice of reception.

Example:

C: RETR 2

The following message is sent if the Cmail server sends the notice of reception:

S: +OK 200 octets

S: <the Cmail server sends the notice of reception including the challenge>

S: .

The following message is sent when the server cannot send the notice of reception:

503 Error: impossible to send Notice-of-reception

The Cmail server finds in the notice of deposit the node `Entity` related to the recipient. Then the Cmail server copies this node in the notice of reception and removes the content of the `Response` node included in the `Entity` node.

Example, a node in the notice of deposit:

```
<Entity EmailAddress="john.doe@example.org" Type="to">
  <SecretQuestion>
    <Request RandomNumber="30987497498789739837"/>
   <Response AlgorithmIdentifier="2.16.840.1.101.3.4.2.1"
Encoding="base64">5mYZWhtl0yxBa/wl7VLiiQ=</response>
  </SecretQuestion>
  <CipherEnvelopeKey Algorithm="AES" CipheredKey="RSA" Encoding="base64-DER"
KeySize="256">UjBg…b1PHDOOM4IFnTpzHn9TQ==</cipherEnvelopeKey>
  <Certificate
Encoding="base64">MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM…sdjn7VDBlb+WS10j2rJcAHHsUyr…
/gy7</Certificate>
</Entity>
```

NOTE 1 – This challenge could use ASN.1 DER encoding.

And the same node copied in the notice of reception:

```
<Entity EmailAddress="john.doe@example.org" Type="to">
  <SecretQuestion>
    <Request RandomNumber="30987497498789739837"/>
   <Response AlgorithmIdentifier="2.16.840.1.101.3.4.2.1" Encoding="base64" />
  </SecretQuestion>
  <CipherEnvelopeKey Algorithm="AES" CipheredKey="RSA" Encoding="base64-DER"
KeySize="256">UjBg…b1PHDOOM4IFnTpzHn9TQ==</cipherEnvelopeKey>
  <Certificate
Encoding="base64">MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM…sdjn7VDBlb+WS10j2rJcAHHsUyr…
/gy7</Certificate>
</Entity>
```

NOTE 2 – This challenge could use ASN.1 DER encoding.

9.3 Challenge response and recipient and server signed notice of reception

The recipient shall:

1. decode the received notice of reception;
2. retrieve the RSCK;
3. compute challenge response;
4. sign the server signed notice of reception using its own private key;
5. encode the result in base64; and
6. transmit it to the Cmail server using:

CHLG RESP <challenge response and recipient and server signed notice of reception>

The recipient deciphers the message as follows:

```
<Entity EmailAddress="john.doe@example.org" Type="to">
  <SecretQuestion>
    <Request RandomNumber="30987497498789739837"/>
   <Response AlgorithmIdentifier="2.16.840.1.101.3.4.2.1" Encoding="base64"></response>
  </SecretQuestion>
  <CipherEnvelopeKey Algorithm="AES" CipheredKey="RSA" Encoding="base64-DER"
KeySize="256">UjBg…b1PHDOOM4IFnTpzHn9TQ==</cipherEnvelopeKey>
  <Certificate
Encoding="base64">MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AM…sdjn7VDBlb+WS10j2rJcAHHsUyr…
/gy7</Certificate>
</Entity>
```

NOTE - This challenge could use ASN.1 DER encoding.

The recipient recovers RSCK using his private key by deciphering the content of the node **`CipherEnvelopeKey`**. Then the recipient concatenates **`RandomNumber`** and RSCK, hashes it using the defined **`AlgorithmIdentifier`**, and obtains the result of the **`SecretQuestion`**.

The recipient copies this result in the signed notice of reception, signs it and sends it to the Cmail server.

9.4 ENVELOPE

If the challenge is OK, the Cmail server sends the ENVELOPE in the same way as the result of the command RETR. The recipient now has the message and the key to open it.
The following message is sent when the server cannot send the ENVELOPE:

503 Error: impossible to send ENVELOPE

9.5 Recipient and server signed notice of reception between Cmail servers (optional)

This message is only sent if the sender and the recipient are attached to different Cmail servers.

SEND NORP <base64 encoded Recipient and server signed notice of reception>

9.6 Recipient and server signed notice of reception

This message is only sent if the sender and the recipient are attached to different Cmail servers.

SEND NORP <base64 encoded Recipient and server signed notice of reception>

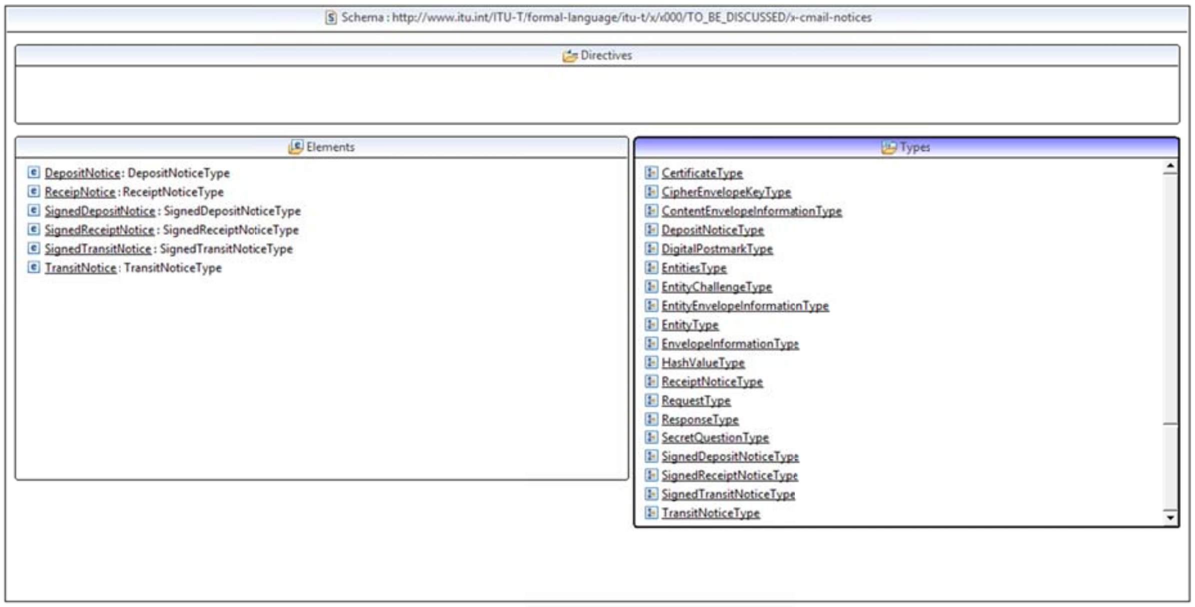
Annex A

Notices in XML schema definition (XSD)

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

This Annex provides the specification of notices using the XML schema definition (XSD) as specified in [XSD]. An instance of communication is encoded in XML as specified in [XML] and shall be in according with to the XSD specifications given in this Annex.

A.1 XSD overview



Schema : http://www.itu.int/ITU-T/formal-language/itu-t/x/x000/TO_BE_DISCUSSED/x-cmail-notices

Directives

Elements

- DepositNotice : DepositNoticeType
- ReceipNotice : ReceiptNoticeType
- SignedDepositNotice : SignedDepositNoticeType
- SignedReceiptNotice : SignedReceiptNoticeType
- SignedTransitNotice : SignedTransitNoticeType
- TransitNotice : TransitNoticeType

Types

- CertificateType
- CipherEnvelopeKeyType
- ContentEnvelopeInformationType
- DepositNoticeType
- DigitalPostmarkType
- EntitiesType
- EntityChallengeType
- EntityEnvelopeInformationType
- EntityType
- EnvelopeInformationType
- HashValueType
- ReceiptNoticeType
- RequestType
- ResponseType
- SecretQuestionType
- SignedDepositNoticeType
- SignedReceiptNoticeType
- SignedTransitNoticeType
- TransitNoticeType

Figure A.1 – Elements and type list

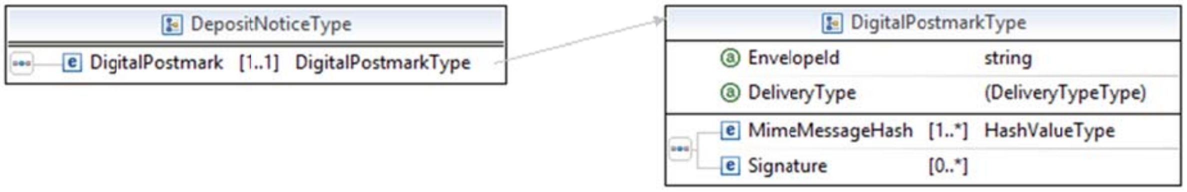

DepositNoticeType
DigitalPostmark [1..1] DigitalPostmarkType
DigitalPostmarkType
EnvelopeId string
DeliveryType (DeliveryTypeType)
MimeMessageHash [1..*] HashValueType
Signature [0..*]

Figure A.2 – Notice of deposit

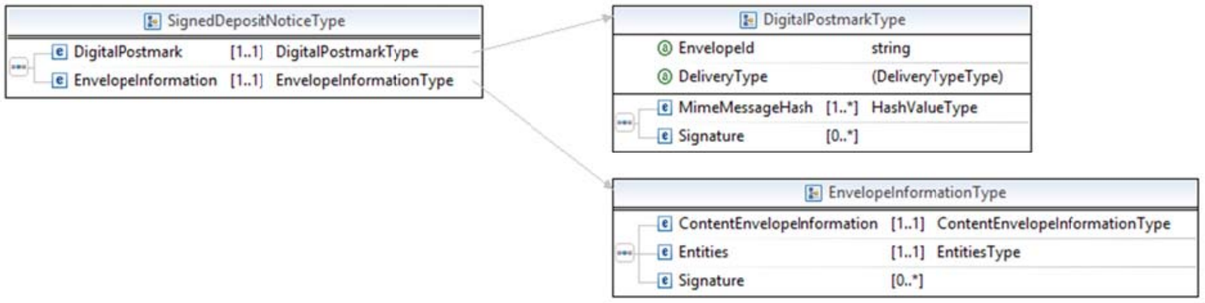

SignedDepositNoticeType
DigitalPostmark [1..1] DigitalPostmarkType
EnvelopeInformation [1..1] EnvelopeInformationType
DigitalPostmarkType
EnvelopeId string
DeliveryType (DeliveryTypeType)
MimeMessageHash [1..*] HashValueType
Signature [0..*]
EnvelopeInformationType
ContentEnvelopeInformation [1..1] ContentEnvelopeInformationType
Entities [1..1] EntitiesType
Signature [0..*]

Figure A.3 – Signed notice of deposit

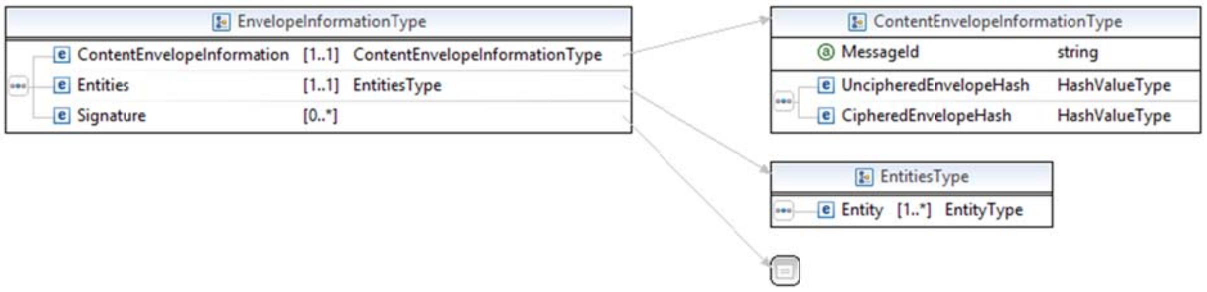

EnvelopeInformationType
ContentEnvelopeInformation [1..1] ContentEnvelopeInformationType
Entities [1..1] EntitiesType
Signature [0..*]
ContentEnvelopeInformationType
MessageId string
UncipheredEnvelopeHash HashValueType
CipheredEnvelopeHash HashValueType
EntitiesType
Entity [1..*] EntityType

Figure A.4 – Envelope information type

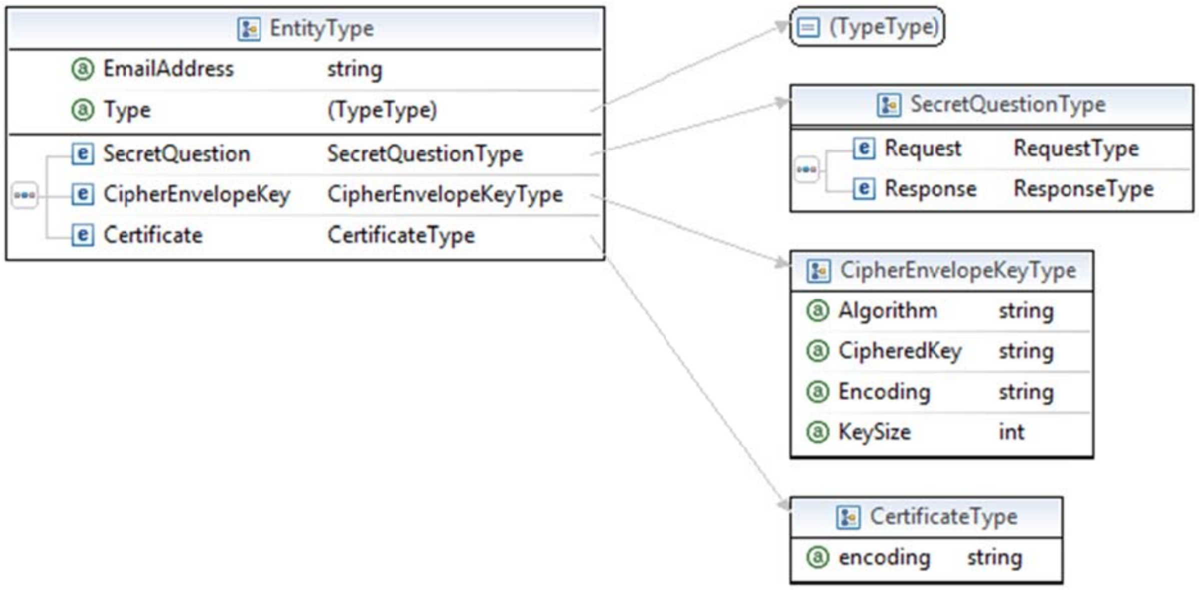


Figure A.5 – Entity type

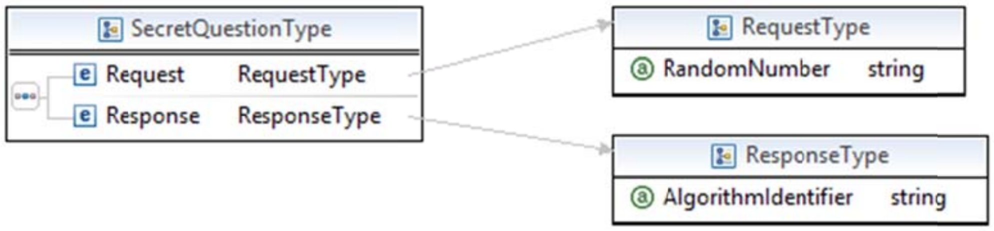


Figure A.6 – Challenge

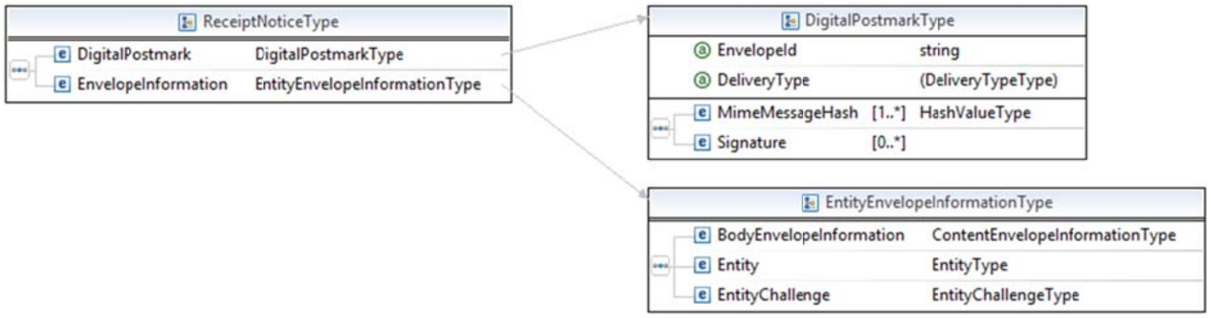

ReceiptNoticeType
DigitalPostmark DigitalPostmarkType
EnvelopeInformation EntityEnvelopeInformationType
DigitalPostmarkType
EnvelopeId string
DeliveryType (DeliveryTypeType)
MimeMessageHash [1..*] HashValueType
Signature [0..*]
EntityEnvelopeInformationType
BodyEnvelopeInformation ContentEnvelopeInformationType
Entity EntityType
EntityChallenge EntityChallengeType

Figure A.7 – Notice of reception

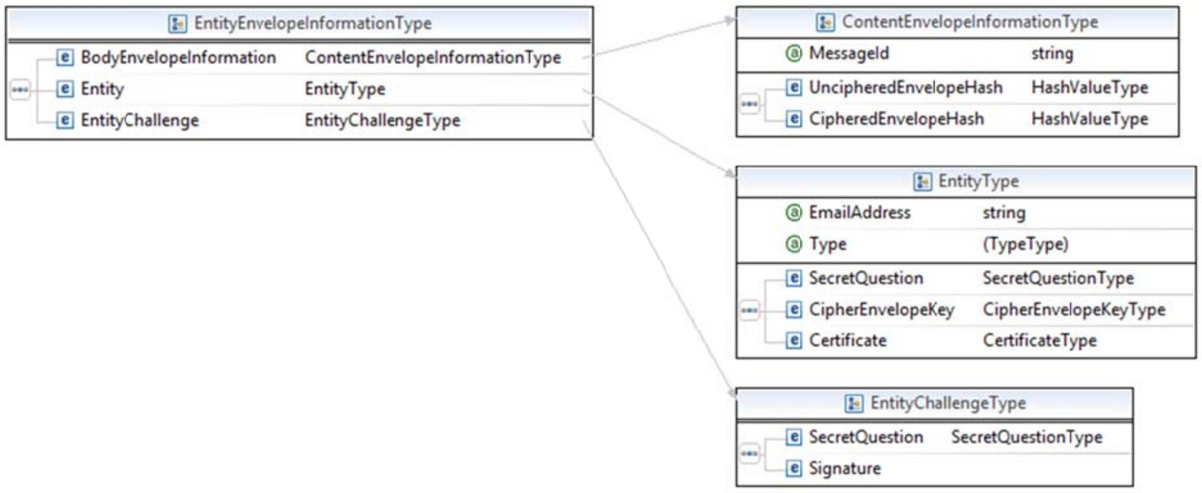

EntityEnvelopeInformationType
BodyEnvelopeInformation ContentEnvelopeInformationType
Entity EntityType
EntityChallenge EntityChallengeType
ContentEnvelopeInformationType
MessageId string
UncipheredEnvelopeHash HashValueType
CipheredEnvelopeHash HashValueType
EntityType
EmailAddress string
Type (TypeType)
SecretQuestion SecretQuestionType
CipherEnvelopeKey CipherEnvelopeKeyType
Certificate CertificateType
EntityChallengeType
SecretQuestion SecretQuestionType
Signature

Figure A.8 – Recipient's answer to challenge

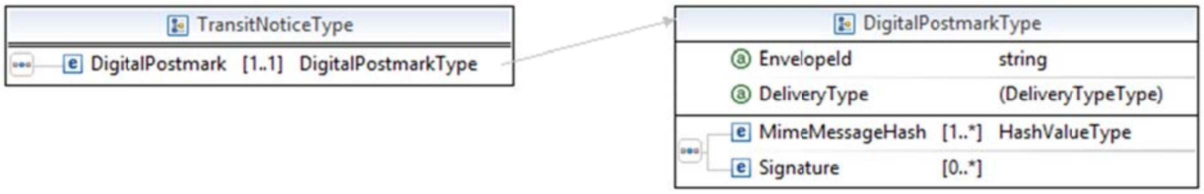

TransitNoticeType
DigitalPostmark [1..1] DigitalPostmarkType
DigitalPostmarkType
EnvelopeId string
DeliveryType (DeliveryTypeType)
MimeMessageHash [1..*] HashValueType
Signature [0..*]

Figure A.9 – Notice of transit

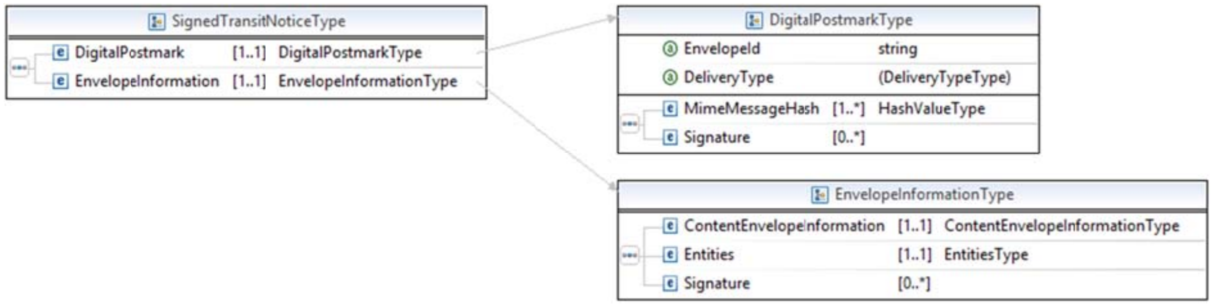

SignedTransitNoticeType
DigitalPostmark [1..1] DigitalPostmarkType
EnvelopeInformation [1..1] EnvelopeInformationType
DigitalPostmarkType
EnvelopeId string
DeliveryType (DeliveryTypeType)
MimeMessageHash [1..*] HashValueType
Signature [0..*]
EnvelopeInformationType
ContentEnvelopeInformation [1..1] ContentEnvelopeInformationType
Entities [1..1] EntitiesType
Signature [0..*]

Figure A.10 – Signed notice of transit

A.2 Formal specification of notices in XSD

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.itu.int/xml-namespace/itu-t/x.1341/x-cmail-notices"
     elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://www.itu.int/xml-namespace/itu-t/x.1341/x-cmail-notices"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

  <import namespace="http://www.w3.org/2009/xmldsig11#"
    schemaLocation="http://www.w3.org/TR/xmldsig-core1/xmldsig11-schema.xsd" />
  <import namespace="http://www.w3.org/2009/xmldsig-properties"
    schemaLocation="http://www.w3.org/TR/xmldsig-properties/xmldsig-properties.xsd" />

  <import schemaLocation=http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd
    namespace="http://www.w3.org/2000/09/xmldsig#"></import>

  <element name="DepositNotice" type="tns:DepositNoticeType"></element>
  <element name="SignedDepositNotice" type="tns:SignedDepositNoticeType"></element>
  <element name="TransitNotice" type="tns:TransitNoticeType"></element>
  <element name="SignedTransitNotice" type="tns:SignedTransitNoticeType"></element>
```

```
<element name="ReceipNotice" type="tns:ReceiptNoticeType"></element>
<element name="SignedReceiptNotice" type="tns:SignedReceiptNoticeType"></element>

<complexType name="DigitalPostmarkType">
  <sequence>
    <element name="MimeMessageHash" type="tns:HashValueType"
      maxOccurs="unbounded" minOccurs="1">
    </element>
    <element name="Signature" type="ds:SignatureType"
      maxOccurs="unbounded" minOccurs="0">
    </element>
  </sequence>
  <attribute name="EnvelopeId" type="string" use="required"></attribute>
  <attribute name="DeliveryType" use="required">
    <simpleType>
      <restriction base="string">
        <enumeration value="CertifiedMail"></enumeration>
      </restriction>
    </simpleType>
  </attribute>
</complexType>

<complexType name="EnvelopeInformationType">
  <sequence>
    <element name="ContentEnvelopeInformation"
      type="tns:ContentEnvelopeInformationType" maxOccurs="1" minOccurs="1">
    </element>
    <element name="Entities" type="tns:EntitiesType"
      maxOccurs="1" minOccurs="1">
    </element>
    <element name="Signature" type="ds:SignatureType"
      maxOccurs="unbounded" minOccurs="0">
    </element>
  </sequence>
</complexType>

<complexType name="ContentEnvelopeInformationType">
  <sequence>
    <element name="UncipheredEnvelopeHash" type="tns:HashValueType"></element>
    <element name="CipheredEnvelopeHash" type="tns:HashValueType"></element>
  </sequence>
  <attribute name="MessageId" type="string"></attribute>
</complexType>

<complexType name="SecretQuestionType">
  <sequence>
    <element name="Request" type="tns:RequestType"></element>
    <element name="Response" type="tns:ResponseType"></element>
  </sequence>
</complexType>

<complexType name="EntityType">
  <sequence>
    <element name="SecretQuestion" type="tns:SecretQuestionType"></element>
    <element name="CipherEnvelopeKey"
      type="tns:CipherEnvelopeKeyType">
    </element>
    <element name="Certificate" type="tns:CertificateType"></element>
  </sequence>
  <attribute name="EmailAddress" type="string" use="required">
    <annotation>
      <documentation>Email address has to be in RFC 822format</documentation>
    </annotation></attribute>
  <attribute name="Type" use="required">
    <simpleType>
      <restriction base="string">
        <enumeration value="from"></enumeration>
        <enumeration value="to"></enumeration>
        <enumeration value="cc"></enumeration>
        <enumeration value="transit"></enumeration>
      </restriction>
    </simpleType>
  </attribute>
</complexType>

<complexType name="CipherEnvelopeKeyType">
  <attribute name="Algorithm" type="string"></attribute>
  <attribute name="CipheredKey" type="string"></attribute>
  <attribute name="Encoding" type="string"></attribute>
  <attribute name="KeySize" type="int"></attribute>
</complexType>

<complexType name="CertificateType">
  <attribute name="encoding" type="string"></attribute>
</complexType>

<complexType name="EntitiesType">
  <sequence>
    <element name="Entity" type="tns:EntityType"
      maxOccurs="unbounded" minOccurs="1">
    </element>
  </sequence>
</complexType>

<complexType name="SignedDepositNoticeType">
  <sequence>
    <element name="DigitalPostmark" type="tns:DigitalPostmarkType"
      maxOccurs="1" minOccurs="1">
    </element>
    <element name="EnvelopeInformation"
      type="tns:EnvelopeInformationType" maxOccurs="1" minOccurs="1">
    </element>
  </sequence>
</complexType>

<complexType name="DepositNoticeType">
  <sequence>
    <element name="DigitalPostmark" type="tns:DigitalPostmarkType"
      maxOccurs="1" minOccurs="1">
    </element>
  </sequence>
</complexType>

<complexType name="TransitNoticeType">
  <sequence>
    <element name="DigitalPostmark" type="tns:DigitalPostmarkType"
      maxOccurs="1" minOccurs="1">
    </element>
  </sequence>
</complexType>

<complexType name="SignedTransitNoticeType">
  <sequence>
    <element name="DigitalPostmark" type="tns:DigitalPostmarkType"
      maxOccurs="1" minOccurs="1">
    </element>
    <element name="EnvelopeInformation"
      type="tns:EnvelopeInformationType" maxOccurs="1" minOccurs="1">
    </element>
  </sequence>
</complexType>

<complexType name="ReceiptNoticeType">
  <sequence>
    <element name="DigitalPostmark"
      type="tns:DigitalPostmarkType">
      </element>
      <element name="EnvelopeInformation"
        type="tns:EntityEnvelopeInformationType">
      </element>
    </sequence>
  </complexType>

  <complexType name="SignedReceiptNoticeType">
    <sequence>
      <element name="DigitalPostmark"
        type="tns:DigitalPostmarkType">
      </element>
      <element name="EnvelopeInformation"
        type="tns:EntityEnvelopeInformationType">
      </element>
    </sequence>
  </complexType>

  <complexType name="HashValueType">
    <attribute name="AlgorithmOID">
      <simpleType>
        <restriction base="string">
          <enumeration value="1.3.14.3.2.26"></enumeration>
          <enumeration value="2.16.840.1.101.3.4.2.1"></enumeration>
        </restriction>
      </simpleType>
    </attribute>
  </complexType>

  <complexType name="EntityEnvelopeInformationType">
    <sequence>
      <element name="BodyEnvelopeInformation" type="tns:ContentEnvelopeInformationType">
      </element>
      <element name="Entity" type="tns:EntityType"></element>
      <element name="EntityChallenge" type="tns:EntityChallengeType"></element>
    </sequence>
  </complexType>

  <complexType name="EntityChallengeType">
    <sequence>
      <element name="SecretQuestion" type="tns:SecretQuestionType"></element>
      <element name="Signature" type="ds:SignatureType"></element>
    </sequence>
  </complexType>

  <complexType name="RequestType">
    <attribute name="RandomNumber" type="string"></attribute>
  </complexType>

  <complexType name="ResponseType">
    <attribute name="AlgorithmIdentifier" type="string"></attribute>
  </complexType>

</schema>
```

Annex B

Notices in ASN.1

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

This Annex provides the specification of notes in the abstract syntax notation one (ASN.1) as specified in [X.680]. The notices may be encoded using the ASN.1 distinguished encoding rules (DER) as specified in [X.690] or using the extended XML encoding rules (EXTENDED-XER) as specified in [X.693]. In the latter case, the XML resulting from this encoding is identical to the XML generated according to the XDS as specified in Annex A.

```
CMAIL {itu-t(0) recommendation(0) x(24) cmail(1341) asn1Module(1) cmail(1)}
DEFINITIONS AUTOMATIC TAGS ::=
BEGIN

IMPORTS String
     FROM XSDv2 {joint-iso-itu-t asn1(1) specification(0) modules(0)
          xsd-module(2) version2(2)};

DepositNotice        ::= DepositNoticeType

SignedDepositNotice  ::= SignedDepositNoticeType

TransitNotice        ::= TransitNoticeType

SignedTransitNotice  ::= SignedTransitNoticeType

ReceiptNotice        ::= ReceiptNoticeType

SignedReceiptNotice  ::= SignedReceiptNoticeType

DigitalPostmarkType  ::= SEQUENCE {
     mimeMessageHash SEQUENCE (SIZE(1..MAX)) OF
          mimeMessageHash HashValueType,
     signature  SEQUENCE (SIZE(0..MAX)) OF
          signature  SignatureType,
     envelopeId String,
     deliveryType    ENUMERATED {
          certifiedMail,
          ...
          }
     }

EnvelopeInformationType    ::= SEQUENCE {
     contentEnvelopeInformation ContentEnvelopeInformationType,
     entities              EntitiesType,
     signature       SEQUENCE (SIZE(0..MAX)) OF
          signature SignatureType
     }

ContentEnvelopeInformationType  ::= SEQUENCE {
     uncipheredEnvelopeHash     HashValueType,
     cipheredEnvelopeHash HashValueType,
     messageId       String
     }

SecretQuestionType   ::= SEQUENCE {
     request         RequestType,
     response   ResponseType
     }

EntityType ::= SEQUENCE {
     secretQuestion       SecretQuestionType,
     cipheredEnvelopeKey  CipheredEnvelopeKeyType,
     certificate          CertificateType,
     emailAddress         String
          (CONSTRAINED BY
          {-- "Email address has to be in IETF RFC 822 format --}),
     type ENUMERATED {
          from,
          to,
          cc,
          transit
          }
     }

CipheredEnvelopeKeyType   ::= SEQUENCE {
     algorithm  String,
     ciphereredKey   String,
     encoding   String,
     keySize         String
     }

CertificateType ::= SEQUENCE {
     encoding   String
     }

EntitiesType    ::= SEQUENCE {
     entity          SEQUENCE(SIZE(1..MAX)) OF entity EntityType
     }

SignedDepositNoticeType   ::= SEQUENCE {
     digitalPostmark      DigitalPostmarkType,
     envelopeInformation  EnvelopeInformationType
     }

DepositNoticeType    ::= SEQUENCE {
     digitalPostmark      DigitalPostmarkType
     }

TransitNoticeType    ::= SEQUENCE {
     digitalPostmark      DigitalPostmarkType
     }

SignedTransitNoticeType   ::= SEQUENCE {
     digitalPostmark      DigitalPostmarkType,
     envelopeInformation  EnvelopeInformationType
     }

ReceiptNoticeType    ::= SEQUENCE {
     operatorPostmark     DigitalPostmarkType
     }

SignedReceiptNoticeType   ::= SEQUENCE {
     operatorPostmark     DigitalPostmarkType,
     envelopeInformation  EntityEnvelopeInformationType
     }

HashValueType   ::= SEQUENCE {
     algorithmOID    ENUMERATED {
          sha-1,
          sha-256
          }
     }

EntityEnvelopeInformationType   ::= SEQUENCE {
     bodyEnvelopeInformation   ContentEnvelopeInformationType,
     entity               EntityType,
     entityChallenge      EntityChallengeType
     }

EntityChallengeType  ::= SEQUENCE {
     secretQuestion _SecretQuestionType,
     signature  SignatureType
     }

RequestType     ::= SEQUENCE {
     randomNumer     String
     }

ResponseType    ::= SEQUENCE {
     algorithmIdentifier  String
     }

SignatureType   ::= String

ENCODING-CONTROL XER
     GLOBAL-DEFAULTS MODIFIED-ENCODINGS
     [NAME AS CAPITALIZED] DigitalPostmarkType.mimeMessageHash
     [UNTAGGED] DigitalPostmarkType.mimeMessageHash
     [NAME AS CAPITALIZED] DigitalPostmarkType.signature.*
     [UNTAGGED] DigitalPostmarkType.signature
     [NAME AS CAPITALIZED] DigitalPostmarkType.envelopeId
     [ATTRIBUTE] DigitalPostmarkType.envelopeId
     [NAME AS CAPITALIZED] DigitalPostmarkType.deliveryType
     [ATTRIBUTE] DigitalPostmarkType.deliveryType
     [TEXT AS CAPITALIZED] DigitalPostmarkType.delivetyType:certifiedMail
     [NAME AS CAPITALIZED] EnvelopeInformationType.contentEnvelopeInformation
     [NAME AS CAPITALIZED] EnvelopeInformationType.entities
     [NAME AS CAPITALIZED] EnvelopeInformationType.signature
     [UNTAGGED] EnvelopeInformationType.signature
     [NAME AS CAPITALIZED]
          ContentEnvelopeInformationType.uncipheredEnvelopeHash
     [NAME AS CAPITALIZED]
          ContentEnvelopeInformationType.cipheredEnvelopeHash
     [NAME AS CAPITALIZED] ContentEnvelopeInformationType.messageId
     [ATTRIBUTE] ContentEnvelopeInformationType.messageId
     [NAME AS CAPITALIZED] SecretQuestionType.request
     [NAME AS CAPITALIZED] SecretQuestionType.response
     [NAME AS CAPITALIZED] EntityType.secretQuestion
     [NAME AS CAPITALIZED] EntityType.cipheredEnvelopeKey
     [NAME AS CAPITALIZED] EntityType.certificate
     [NAME AS CAPITALIZED] EntityType.emailAddress
     [ATTRIBUTE] EntityType.emailAddress
     [NAME AS CAPITALIZED] EntityType.type
     [ATTRIBUTE] EntityType.type
     [NAME AS CAPITALIZED] CipheredEnvelopeKeyType.algorithm
     [ATTRIBUTE] CipheredEnvelopeKeyType.algorithm
     [NAME AS CAPITALIZED] CipheredEnvelopeKeyType.cipheredKey
     [ATTRIBUTE] CipheredEnvelopeKeyType.cipheredKey
     [NAME AS CAPITALIZED] CipheredEnvelopeKeyType.encoding
     [ATTRIBUTE] CipheredEnvelopeKeyType.encoding
     [NAME AS CAPITALIZED] CipheredEnvelopeKeyType.keysize
     [ATTRIBUTE] CipheredEnvelopeKeyType.keysize
     [NAME AS CAPITALIZED] CertificateType.encoding
     [ATTRIBUTE] CertificateType.encoding
     [UNTAGGED] EntitiesType.entity
     [NAME AS CAPITALIZED] EntitiesType.entity.*
     [NAME AS CAPITALIZED] SignedDepositNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] SignedDepositNoticeType.envelopeInformation
     [NAME AS CAPITALIZED] DepositNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] TransitNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] SignedTransitNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] SignedTransitNoticeType.envelopeInformation
     [NAME AS CAPITALIZED] ReceiptNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] SignedReceiptNoticeType.digitalPostmark
     [NAME AS CAPITALIZED] SignedReceiptNoticeType.envelopeInformation
     [NAME AS CAPITALIZED] HashValueType.algorithmOID
     [ATTRIBUTE] HashValueType.algorithmOID
     [TEXT AS "1.3.14.3.2.26"] HashValueType.algorithmOID:sha-1
     [TEXT AS "2.16.840.1.101.3.4.2.1"] HashValueType.algorithmOID:sha-256
     [NAME AS CAPITALIZED]
          EntityEnvelopeInformationType.BodyEnvelopeInformation
     [NAME AS CAPITALIZED]
          EntityEnvelopeInformationType.entityChallenge
     [NAME AS CAPITALIZED] EntityChallengeType.secretQuestion
     [NAME AS CAPITALIZED] EntityChallengeType.signature
     [NAME AS CAPITALIZED] RequestType.randomNumber
     [ATTRIBUTE] RequestType.randomNumber
     [NAME AS CAPITALIZED] ResponseType.algorithmIdentifier
     [ATTRIBUTE] ResponseType.algorithmIdentifier

END
```

Annex C

Requirements on public-key infrastructure components

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

C.1 Introduction

This annex provides requirements on public-key certificates issued to Cmail servers and clients.

C.2 Cmail server end-entity public-key certificates

An end-entity public-key certificate issued to a Cmail server shall have the following content:

a) The version 3 shall be specified.

b) The CA shall generate non-sequential serial numbers.

c) The subject field shall hold a directory distinguished name with a single component using the **`dnsName`** attribute type as defined in [ITU-T X.520]. The value shall be a registered DNS name.

d) The Subject alternative name extension shall be present with two elements:

- the **`rfc822Name`** alternative shall be taken for one of the elements and shall be the e-mail address of the administrator of the Cmail server.
- the **`directoryName`** alternative shall be taken for the other element and shall hold a distinguished name with the following components:
  - **`countryName`** shall be present and shall hold the three-letter code (alpha-3) of [ISO 3166-1].
  - **`organizationName`** shall be present and shall hold the ~~legal~~ trusted name of the organisation managing the Cmail server;
  - **`streetAddress`** shall be present and shall hold the street name and the house number;
  - **`localityName`**: shall be present and shall hold the name of the locality;
  - **`stateOrProvinceName`** shall be present if necessary for unique identification. Otherwise, it shall be absent.
  - **`postalCode`** shall be present and shall hold the postal code for the location.

e) The **`certificatePolicies`** extension shall be present and shall at least hold the object identifier **`{itu-t(0) recommendation(0) x(24) cmail(1341) certificatePolicy(2) cmailServer(1)}`** to signal that the public-key certificate is issued according to this Recommendation.

C.3 Cmail client end-entity public-key certificates

An end-entity public-key certificate issued to a Cmail client shall have the following content:

a) The version 3 shall be specified.

b) The CA shall generate non-sequential serial numbers.

c) The subject field shall hold a directory distinguished name with components as follows:

- **`surname`** shall be present if the client is an individual, but shall be absent if the client is an organisation.
- **`givenName`** shall be present if the surname is present. Otherwise, it shall be absent.

- **`initials`** may be present if **`surname`** is present. Otherwise, it shall be absent.
- **`generationQualifier`** may be present if **`surname`** is present. Otherwise, it shall be absent.
- **`organizationName`** shall be present if the client is not a residential person. Otherwise, it shall be absent. If present, it shall hold the ~~legal~~ trusted name of the organisation to which the client belongs.
- **`streetAddress`** shall be present and shall hold the street name and the house number.
- **`localityName`** shall be present and shall hold the name of the locality.
- **`stateOrProvinceName`** shall be present if necessary for unique identification. Otherwise, is shall be absent.
- **`postalCode`** shall be present and shall hold the postal code for the location.
- **`countryCode3c`** shall be present and shall hold the three-letter code (alpha-3) of [ISO 3166-1].

d) The **`subjectAltName`** extension shall be present. It shall contain one element as indicated below:

- **`rfc822Name`** shall hold the e-mail address of administrator of the Cmail server.

e) The certificatePolicies extension shall be present and shall at least hold the object identifier **`{itu-t(0) recommendation(0) x(24) cmail(1341) certificatePolicy(2) cmailClient(2)}`** to signal that the public-key certificate is issued according to this Recommendation.

C.4 Information validation requirements

Before issuing a public-key certificate the issuer shall verify:

a) Verify that the subject (applicant) is the registered holder of the domain name to be included in the public-key certificate.

b) Verify subject's physical existence.

c) Verify the subject's operational existence (business activity).

d) Verify that the subject is a ~~legally~~ trusted recognized entity.

e) Verify the name and address information to be placed in a public-key certificate.

f) Verify that an **`organizationName`** to be entered into a public-key certificate is a ~~legal~~ trusted and recognized name identifying the subject.

Annex D

Requirements on transport layer security (TLS)

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

[IETF RFC 5246] or later shall be supported.

In the negotiation, neither the Cmail server nor the client shall accept a connection where there is an attempt to negotiate a TLS version earlier than TLS 1.2.

An implementation shall support the following cipher suite:

- TLS_DH_RSA_WITH_AES_256_CBC_SHA256

Annex E

Object identifiers defined in this Recommendation

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

This Recommendation defines the following object identifiers:

a) object identifier associated to the ASN.1 module:

```
{itu-t recommendation(0) x(24) cmail(1341) asn1module(0) cmail(1)}
```

b) object identifier used by the certificatePolicies extension of a cmail Server:

```
{itu-t recommendation(0) x(24) cmail(1341) certificateProfile(2) cmailServer(1)}
```

c) object identifier aused by the certificatePolicies extension of a cmail Client:

```
{itu-t recommendation(0) x(24) cmail(1341) certificateProfile(2) cmailClient(2)}
```

Appendix I

Envelope and notices format

(This appendix does not form an integral part of this Recommendation.)

This Appendix provides example of notice encodings

I.1 Notice of deposit

The notice of deposit contains information about the sender, the envelope, and is co-signed by the Cmail server and the sender.

It is an evidence of deposit for the sender who can use it in case of litigation.

The formal specification of the notice of deposit can be found in Annex A.

Example: file "1373360283931.deposit.notice"

```
Received: from localhost ([127.0.0.1])
        by begmeil
        with SMTP (SubEthaSMTP null) id HIWV8HF9
        for laura.prin@legalbox.com;
        Tue, 09 Jul 2013 10:58:14 +0200 (CEST)
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=depositNotice.xml
```

```
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiIHN0YW5kYWxvbmU9Im5vIj8+Cjxs
ZXR0ZXJEZXBvc2l0UG9zdG1hcms+CiAgPG9wZXJhdG9yUG9zdG1hcms+CiAgICA8ZW52ZWxvcElk
bG9wZWQtc2lnbmF0dXJlIi8+CiAgICAgICAgICA8L1RyYW5zZm9ybXM+CiAgICAgICAgICA8RGln
...
ICAgPFJTQUtleVZhbHVlPgogICAgICAgICAgICA8TW9kdWx1cz5tMkFSUURXUGJBMmgvMzJEQWs4
ICAgICAgICAgIDxFeHBvbmVudD5BUUFCPC9FeHBvbmVudD4KICAgICAgICAgIDwvUlNBS2V5VmFs
dWU+CiAgICAgICAgPC9LZXlWYWx1ZT4KICAgICAgPC9LZXlJbmZvPgogICAgPC9TaWduYXR1cmU+
CiAgPC9lbnZlbG9wSW5mb3JtYXRpb24+CjwvbGV0dGVyRGVwb3NpdFBvc3RtYXJrPgo=
```

I.2 Notice of reception

The notice of reception contains information about the sender, the envelope, the challenge to open the envelope, and is co-signed by the Cmail server and the recipient.

It is an evidence of reception for the sender who can use it in case of litigation.

The formal specification of the notice of reception can be found in Annex A.

Example: file "1373360283931.laura.prin@legalbox.com.receipt.notice"

```
Received: from begmeil get hostname ([127.0.0.1])
        by localhost
        with SMTP (LegalBox POP Server v1.0) id HIWX27L5
        for laura.prin@legalbox.com;
        Tue, 09 Jul 2013 11:49:01 +0200 (CEST)
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=receiptNotice.xml
```

```
PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiIHN0YW5kYWxvbmU9Im5vIj8+Cjxs
ZXR0ZXJEZXBvc2l0UG9zdG1hcms+CiAgPG9wZXJhdG9yUG9zdG1hcms+CiAgICA8ZW52ZWxvcElk
PjEzNzMzNjAyODM5MzE8L2VudmVsb3BJZD4KICAgIDxkZWxpdmVyeU1vZGU+Y2VydGlmaWVkTGV0
dGVyPC9kZWxpdmVyeU1vZGU+CiAgICA8bWltZU1lc3NhZ2VIYXNoPgogICAgICA8c2hhMT5hNTVk
ZDhmYWU0Mzg2M2VmYWRmMWY3ZjM3MmEwYmU1MmEwMGRhYTFkPC9zaGExPgogICAgPC9taW1lTWVz
...
MDkwSDl0NFVkTTdWVU92bjY3WlU2aTJvVSt3b3lGR2tYMDJ3YkVMM2pDYmpJCm5VR1BwUGpoT3Zo
dzNPTy9mYmhKVk13dkM2NXB1MTl1cnA2M05kS0tHNlBuNjZtQkVnUldxZ2cvTVBITmZmWkhrOXFs
WExSSXhETi8Kb0ZnS285RmI0NExlSzBnZ3Vyb1Y2azNicm1TeGM1UnpYVWNxTzdwbldUN0FoNFl6
WXJJUHdYL1hjS1VqbXYxZi9JZjQ5VHVnWGtLcgpodklyOG9qUkdQcEdPdlB4cWR5QWNQR1BOUVRY
NFJrc29kSEVWdz09PC9Nb2R1bHVzPgogICAgICAgICAgICA8RXhwb25lbnQ+QVFBQjwvRXhwb25l
bnQ+CiAgICAgICAgICA8L1JTQUtleVZhbHVlPgogICAgICAgIDwvS2V5VmFsdWU+CiAgICAgIDwv
```

```
S2V5SW5mbz4KICAgIDwvU2lnbmF0dXJlPgogIDwvcmVjaXBpZW50Q2hhbGVuZ2U+CjwvbGV0dGVy
RGVwb3NpdFBvc3RtYXJrPgo=
```

I.3 Notice of transit

The notice of deposit contains information about the sender, the envelope, the challenge to open the envelope, and is co-signed by the Cmail servers.

It is an evidence of transit for the sender who can use it in case of litigation.

The formal specification of the notice of transit can be found in Annex A.

Example: file "1373360283931.laura.prin@legalbox.com.receipt.notice"

```
Received: from begmeil get hostname ([127.0.0.1])
        by localhost
        with SMTP (LegalBox POP Server v1.0) id HIWX27L5
        for laura.prin@legalbox.com;
        Tue, 09 Jul 2013 11:49:01 +0200 (CEST)
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=receiptNotice.xml

PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiIHN0YW5kYWxvbmU9Im5vIj8+Cjxs
ZXR0ZXJEZXBvc2l0UG9zdG1hcms+CiAgPG9wZXJhdG9yUG9zdG1hcms+CiAgICA8ZW52ZWxvcElk
PjEzNzMzNjAyODM5MzE8L2VudmVsb3BJZD4KICAgIDxkZWxpdmVyeU1vZGU+Y2VydGlmaWVkTGV0
dGVyPC9kZWxpdmVyeU1vZGU+CiAgICA8bWltZU1lc3NhZ2VIYXNoPgogICAgICA8c2hhMT5hNTVk
ZDhmYWU0Mzg2M2VmYWRmMWY3ZjM3MmEwYmU1MmEwMGRhYTFkPC9zaGExPgogICAgPC9taW1lTWVz
...
MDkwSDl0NFVkTTdWVU92bjY3WlU2aTJvVSt3b3lGR2tYMDJ3YkVMM2pDYmpJCm5VR1BwUGpoT3Zo
dzNPTy9mYmhKVk13dkM2NXB1MTl1cnA2M05kS0tHNlBuNjZtQkVnUldxZ2cvTVBITmZmWkhrOXFs
WExSSXhETi8Kb0ZnS285RmI0NExlSzBnZ3Vyb1Y2azNicm1TeGM1UnpYVWNxTzdwbldUN0FoNFl6
WXJJUHdYL1hjS1VqbXYxZi9JZjQ5VHVnWGtLcgpodklyOG9qUkdQcEdPdlB4cWR5QWNQR1BOUVRY
NFJrc29kSEVWdz09PC9Nb2R1bHVzPgogICAgICAgICAgICA8RXhwb25lbnQ+QVFBQjwvRXhwb25l
bnQ+CiAgICAgICAgICAgICA8L1JTQUtleVZhbHVlPgogICAgICAgICAgIDwvS2V5VmFsdWU+CiAgICAgIDwv
S2V5SW5mbz4KICAgIDwvU2lnbmF0dXJlPgogIDwvcmVjaXBpZW50Q2hhbGVuZ2U+CjwvbGV0dGVy
RGVwb3NpdFBvc3RtYXJrPgo=
```

I.4 ENVELOPE

ENVELOPE is a MIME message containing the e-mail content ciphered by AES encryption.

Example: file "1373360283931.certifiedLetter.msg"

```
Received: from localhost ([127.0.0.1])
        by begmeil
        with SMTP (SubEthaSMTP null) id HIWV8HF9
        for laura.prin@legalbox.com;
        Tue, 09 Jul 2013 10:58:03 +0200 (CEST)
Date: Tue, 9 Jul 2013 10:57:51 +0200 (CEST)
From: david.keller@legalbox.com
To: laura.prin@legalbox.com
Message-ID: proto_cmtp_1373360269856
Subject: =?UTF-8?Q?Bienvenue_=C3=A0_CMTP!?=
MIME-Version: 1.0
Content-Type: multipart/mixed;
      boundary="----=_Part_1_1013939722.1373360271613"

------=_Part_1_1013939722.1373360271613
Content-Type: multipart/mixed;
      boundary="----=_Part_0_2062834323.1373360271584"

------=_Part_0_2062834323.1373360271584
Content-Type: application/octet-stream
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=envelop

RG44gUlyr1A/L+ps0R+yKMUpgPcJACmcRQdLZSMoLnm07gtRataSAWkG5qnc/f5Q

------=_Part_0_2062834323.1373360271584--

------=_Part_1_1013939722.1373360271613--
```

Bibliography

[b-ITU-T X.509] Recommendation ITU-T X.509 (2012) | ISO/IEC 9594-8:2014, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks*.

____