ETSI TS 103 816-3 V1.1.1 (2021-07)

Publicly Available Specification (PAS);
CYBER;
Connecting Products based on MIKEY-SAKKE;
Part 3: One-to-One Messaging

CAUTION

The present document has been submitted to ETSI as a PAS produced by Secure Chorus and approved by the ETSI Technical Committee Cyber Security (CYBER).

Reference
DTS/CYBER-0065-3

Keywords
cyber security, mobile, PAS

ETSI

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

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

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

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

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

Trademarks

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

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Cyber Security (CYBER).

The present document is part 3 of a multi-part deliverable covering Connecting Products based on MIKEY-SAKKE, as identified below:

Part 1: "KMS Certificate Definition";

Part 2: "One-to-One Voice Communication";

Part 3: "One-to-One Messaging";

Part 4: "Group Voice Communication";

Part 5: "Discovery".

Modal verbs terminology

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

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

1 Scope

The present document is intended to specify the interface used for encrypted one-to-one text messages. It is intended for use in connecting products based on Multimedia Internet Keying Sakai-Kasahara Key Encryption (MIKEY-SAKKE) domains and to validate products.

2 References

2.1 Normative references

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

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

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

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

[1] IETF RFC 2045 (November 1996): "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies". N. Freed and N. Borenstein.

[2] IETF RFC 3830 (August 2004): "MIKEY: Multimedia Internet KEYing". J. Arkko et al.

[3] IETF RFC 3966 (December 2004): "The tel URI for Telephone Numbers". H. Schulzrinne.

[4] IETF RFC 4648 (October 2006): "The Base16, Base32, and Base64 Data Encodings". S. Josefsson.

[5] IETF RFC 6120 (March 2011): "Extensible Messaging and Presence Protocol (XMPP): Core". P. Saint-Andre.

[6] IETF RFC 6121 (March 2011): "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence". P. Saint-Andre.

[7] IETF RFC 6507 (February 2012): "Elliptic Curve-Based Certificateless Signatures for Identity-Based Encryption (ECCSI)". M. Groves.

[8] IETF RFC 6508 (February 2012): "Sakai-Kasahara Key Encryption (SAKKE)". M. Groves.

[9] IETF RFC 6509 (February 2012): "MIKEY-SAKKE: Sakai-Kasahara Key Encryption in Multimedia Internet KEYing (MIKEY)". M. Groves.

[10] IETF RFC 7622 (September 2015): "Extensible Messaging and Presence Protocol (XMPP): Address Format". P. Saint-Andre.

[11] ETSI TS 103 816-1: "Publicly Available Specification (PAS); CYBER; Connecting Products based on MIKEY-SAKKE; Part 1: KMS Certificate Definition".

[12] ETSI TS 103 816-2: "Public Available Specification; CYBER; Part 2: One-to-One Voice Communication".

[13] XEP-0134 (December 2004) (Version 1.1): "XMPP Design Guidelines". P. Saint-Andre, XMPP Standards Foundation (XSF).

[14] XEP-0184 (August 2018) (Version 1.4.0): "Message Delivery Receipts". P. Saint-Andre and J. Hildebrand, XMPP Standards Foundation (XSF).

[15] XEP-0234 (June 2019) (Version 0.19.1): "Jingle File Transfer". P. Saint-Andre and L. Stout, XMPP Standards Foundation (XSF).

[16] XEP-0384 (July 2018) (Version 0.3.0): "OMEMO Encryption". A. Straub, XMPP Standards Foundation (XSF).

2.2 Informative references

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

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

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

Not applicable.

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

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

AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
AES-GCM	AES using GCM
ECCSI	Elliptic Curve-based Certificateless Signatures for IBE

NOTE: See IETF RFC 6507 [7]).

GCM	Galois Counter Mode
IBE	Identity Based Encryption
IETF	Internet Engineering Task Force
IQ	Info/Query
IV	Initialization Vector
JID	XMPP address (historically Jabber IDentity)
KMS	Key Management Service
MIKEY	Multimedia Internet KEYing

NOTE: See IETF RFC 3830 [2].

MIKEY-SAKKE Multimedia Internet KEYing using SAkai-Kasahara Key Encryption

NOTE: See IETF RFC 6509 [9].

OMEMO	Multi-End Message and Object encryption (recursive definition)
PRF	Pseudo-Random Function

RFC Request For Comments
SAKKE SAkai-Kasahara Key Encryption

NOTE: See IETF RFC 6508 [8].

SDP Session Description Protocol
SIP Session Initiation Protocol
SRTP Secure Real-time Transport Protocol
SSV Shared Secret Value
TEK Traffic Encryption Key
TLS Transport Layer Security tel URI
URI Uniform Resource Identifier
URL Uniform Resource Locator
XEP XMPP Extension Protocol
XML eXtensible Markup Language
XMPP eXtensible Messaging and Presence Protocol

NOTE: See IETF RFC 6120 [5]).

XSF XMPP Standards Foundation

4 Background

4.1 Messaging

The messaging standard that is assumed by the present document is eXtensible Messaging and Presence Protocol (XMPP) IETF RFC 6120 [5].

Using XMPP, text and other messages are usually sent from the source to the destination via one or more servers. While the message can be encrypted to and from the server, for example using Transport Layer Security (TLS), that does not protect the messages from being read in transit, in particular by parties with access to a server through which the message transited.

The solution to that problem is the end-to-end encryption of messages. Note that this end-to-end encryption will be in addition to any other forms of encryption that are applied - the end-to-end encrypted messages will use the normal means of transport of XMPP messages from one endpoint (as participants in this protocol are referred to) to another endpoint, which may involve methods such as TLS.

End-to-end encryption has its own problems, specifically that with sufficiently strong encryption - which is readily available - the message not only cannot be read by unauthorised parties between the endpoints, it cannot be read by any possible authorized party between the endpoints unless the key material used at the endpoints is made available.

Some systems have such authorized parties, performing an auditing function. This can occur for example in an enterprise system where users are agents of the enterprise. Such systems have a need for end-to-end encryption that is secure against external parties but can be decrypted under carefully managed circumstances by an auditing process.

4.2 Identity-Based Encryption and MIKEY-SAKKE

The present document defines a messaging approach that can be used in such circumstances, using Identity-Based Encryption (IBE). The identity-based cryptography used, and the means by which its per-session public information is transferred, uses MIKEY-SAKKE [9]. Some cryptographic material is created by and distributed from a Key Management Service (KMS); how that creation and distribution is implemented is beyond the scope of the present document. This cryptographic material includes both public domain certificates as defined in ETSI TS 103 816-1 [11] and each endpoint's private key material. There are no individual endpoint certificates. Each endpoint shall use certificates for its cryptographic domain and for the domains of any other endpoints with which it may communicate in either direction.

This identity-based encryption means each endpoint shall have a cryptographic identity. XMPP means each endpoint shall have an identity known as a XMPP address (historically Jabber IDentity) (JID). The relationship between these two identities - in particular that the cryptographic identity can be derived from the JID and the current date - is described in clause 5.1.

The auditing process and KMS are expected to be separate with appropriate safeguards between them. Note that a KMS can provide selective assistance to the auditing process, for example enabling a single message to be decrypted by the auditing process, or broader assistance, for example enabling all communications by an endpoint to be decrypted by the auditing process, but without allowing such messages to be authentically created by the auditing process. Details of the auditing process are beyond the scope of the present document.

4.3 One-to-One Messages

Messaging between endpoints can follow a number of patterns. The simplest of these, and the one defined by the present document, is a one-off message: endpoint A sends a message to endpoint B, and, apart from acknowledgement of the message, that is all. Any subsequent messages, whether sent immediately or after some delay and whether in the same or in the opposite direction, are treated as independent messages; in particular each subsequent message will be separately encrypted using a new message encryption key. A cryptographic session using the present document thus consists of a single message, plus a possible acknowledgement, see clause 4.10; the latter uses the same message encryption key as the message that it acknowledges. A message, but not an acknowledgement, includes both the cryptographic material to establish a session (message) key and the encrypted message information; an acknowledgement contains only the latter.

Higher-level constructs - such as messaging sessions or chatting - can be constructed from the building block defined by the present document, although in some cases not as efficiently as a specialized construction. Such constructions are beyond the scope of the present document.

4.4 Relationship to Voice Standard

A similar problem to that outlined in the previous clause has been addressed for one-to-one voice communication in ETSI TS 103 816-2 [12], which is based on MIKEY-SAKKE [9]. That is an IBE approach that offers additional advantages to those of auditability. Specifically, assuming two appropriately provisioned endpoints, the source can communicate the necessary information that serves as a basis for encryption between the endpoints by sending a single signalling message to the destination. All that the source needs, in addition to standard XMPP requirements, is the cryptographic identity of the destination, which is available as defined in clause 5.1, and the domain certificates for both endpoints.

The present document uses the same MIKEY-SAKKE messages as defined in ETSI TS 103 816-2 [12] initialization vector (see clause 5.4). Some possible simplifications of those messages are NOT used, in order to maximize commonality between the present document and [12].

Note that using ETSI TS 103 816-2 [12], the MIKEY-SAKKE message is carried within a Session Initiation Protocol (SIP) message using formatting defined by the Session Description Protocol (SDP) and the SIP message uses SIP infrastructure (SIP servers) to ensure that the MIKEY-SAKKE message reaches its destination and is acknowledged. None of those mechanisms are used by the present document, which instead includes the MIKEY-SAKKE message within the XMPP message that it defines the encryption of, see clause 5.4.

Note also that the relationship between the Shared Secret Value (SSV) that is transferred - encrypted - by the MIKEY-SAKKE message and the session key used to encrypt payload data is also different in the present document, but only by omitting the later stages that are carried out by Secure Real-time Transport Protocol (SRTP) rather than MIKEY when using [12], see clause 5.5.

4.5 XMPP Message Protocol

Most of IETF RFC 6120 [5] is concerned with the protocol aspects of XMPP, how it can use an eXtensible Markup Language (XML) stream to carry the required data, in the form of XML stanzas, from source to destination. That protocol is used unchanged by the present document, although since the present document only carries single messages, significant parts of that protocol will not be used when using the present document.

Using the present document, before sending a message the source replaces the information containing parts of the message, known as stanzas (see clause 5.6) by a single message stanza that includes an encrypted version of that information. The replacement message stanza also includes the information needed to decrypt that information. The message recipient performs the reverse process to recover the original stanzas. The necessary attributes of the original message stanza, including its source and destination identities, are retained unchanged by the replacement message stanza. The present document also includes the means by which files, stored remotely at provided Uniform Resource Locators (URLs), can be attached to a message, see clause 4.8, and the means by which messages can be acknowledged, see clause 4.10.

4.6 XMPP Message Structure

XMPP carries information in the form of stanzas. Three kinds of stanza are defined in clause 8 of IETF RFC 6120 [5], represented by three kinds of XML element: `<message/>`, `<presence/>` and `</iq>`. A message that is encrypted by the present document will always include the first of these (the message stanza) and may include any combination of the other two. For this version of the present document all of those stanzas that are in a message that is to be encrypted shall be provided to the implementation of the present document at the source endpoint for replacement, by a single message stanza, so that the content of all stanzas can be delivered securely to the destination endpoint.

The present document can, in some circumstances, modify the message stanza prior to encryption, see clauses 4.8 and 4.10. The present document does not modify or use the other two stanzas in any way other than encrypting them if present, so that they are delivered unchanged after decryption. However, some comments on the other two stanzas and their potential future use are provided in clause 4.9.

All stanzas are encrypted together by the present document. One of the requirements for XMPP extensions XEP-0134 [13] is that no other kinds of stanza are introduced. In order to ensure this, the encrypted message is presented for transport to the destination endpoint as a single `<message/>` element, with the same attributes as the unencrypted stanza. In addition, although the XML schema in Appendix A.5 of IETF RFC 6120 [5] indicates that it is optional, some XMPP parsers expect there to be a `<body/>` element within a received `<message/>` element. In part for this reason, the encrypted `<message/>` element defined by the present document contains a `<body/>` element, with no attributes. All information to be sent, including both the encrypted message ciphertext and the required keying information (i.e. the MIKEY-SAKKE message), is included in this `<body/>` element.

The present document operates independently of the content of the message, presence and Info/Query (IQ) stanzas, other than possibly adding attachments and/or message receipt requests to the first of these. The present document can thus be used in conjunction with any extensions to or modifications of these contents, as long as these are not required between message encryption and message decryption. Any such extension or modification is beyond the scope of the present document. However, to claim full compliance with the present document the unencrypted message shall be fully XMPP compliant IETF RFC 6120 [5], and any added XML elements shall be appropriately namespace qualified. A recipient of unrecognised information of this form shall ignore that information. Note that all such information will be encrypted and authenticated by the present document.

4.7 Message Content Encryption

The contents of a message are replaced when using the present document by an encryption of that message that only the recipient of that message (or its KMS) can decrypt, using the information in the included MIKEY-SAKKE message and the recipient's provisioned private key and certificates. The encryption is of a concatenation of all stanzas that are present. Note that this will encrypt a form of the whitespace within and between those elements, and on decryption will restore the identical form of whitespace, thus making this process unaffected by XML whitespace issues. The encryption used by the present document is AES using GCM (AES-GCM), an authenticated encryption algorithm, allowing the decrypted message to also be authenticated. The transmitted `<message/>` element includes the attributes of the original `<message/>` element. As these are also included in the encrypted message they are thus also authenticated.

4.8 Attachments

The present document allows a message to include attachments. An attachment is considered to be equivalent to a file. In order to keep message sizes limited, as required by XEP-0134 [13], and because file sizes are potentially unlimited, the approach used by the present document (as well as by other XMPP extensions) is to put the file on a suitable server and include the file's URL in the XMPP message.

A protocol is required to take a user-provided attached file, put it on a suitably negotiated server and provide a URL and other information to include within the XMPP message. This process is outside the scope of the present document, which just assumes that the file is on a suitable server and that the file's URL and any other required information is available.

The present document has an additional consideration, beyond that of simply including the URL in the message, namely that the attached file shall be encrypted. The file could be encrypted specifically for the recipient, but it is possible, and in some cases may be likely, that a given file may be sent to more than one recipient. For scalability, maintaining a single encrypted copy of that file at a single URL is desirable. This also, advantageously, decouples the file encryption from the cryptographic attributes of the message recipient(s). It is thus also assumed that the file is encrypted by an agreed algorithm and that the encryption key and any other required information are also available to the message source.

Attachments are each defined by a `<content/>` element. This is added to the message stanza before it is encrypted as outlined in clause 4.7, thus protecting all of the attachment information. The `<content/>` element is defined in clause 5.10. Multiple `<content/>` elements may be added.

4.9 Presence and IQ

Presence information forms the second stanza of an XMPP message. The purpose of presence information is described in clause 1.3 of IETF RFC 6121 [6] as:

> *The purpose of using such an application is to exchange relatively brief text messages with particular contacts in close to real time -- often relatively large numbers of such messages in rapid succession, in the form of a one-to-one "chat session".*

This use, especially the sending of large numbers of messages, is not a good fit to the present document, and thus the use of presence information is not considered in this version of the present document. However, presence information has the potential to allow a user or device first to report that it supports the present document (and which version of the present document) and second (as an alternative to the use of a JID to carry this information, see clause 5.1) to allow discovery of which domain a device is in, and thus which certificate to use when sending a message to it. The presence stanza may thus be considered further in a future version of the present document. However, although it is not used, the presence stanza is protected by the present document.

The third, IQ, stanza of an XMPP message allows it to contain Info/Query information. This is described in clause 8.2.3 of IETF RFC 6120 [5] as:

> *Info/Query, or IQ, is a "request-response" mechanism … The semantics of IQ enable an entity to make a request of, and receive a response from, another entity.*

Such a request/response mechanism is also not the primary intent of the present document and thus the IQ stanza is also not considered by this version of the present document, except that it also is protected.

4.10 Message Acknowledgement

A message may be acknowledged using the message delivery receipt mechanism defined in clause 5.9. The message receipt is encrypted using the same authenticated encryption, primarily to ensure its authentication in a whitespace safe manner. The encryption key used is the same as for the message, and thus the message receipt does not include a MIKEY-SAKKE message. This means that an endpoint requesting a message receipt shall retain the message key, or the shared secret value used to create it, for long enough for message delivery receipt, or until it decides that no such receipt is forthcoming.

4.11 Other XMPP Specifications

For the present document a compliant implementation shall implement all relevant parts of the XMPP Standards Foundation (XSF) MPP Extension Protocol (XEP) [14] defining message receipts. The present document borrows and extends concepts and elements from the XEP-0234 [15] and XEP-0384 [16], but does not require the implementation of those two non-standard XEPs. In particular the present document is significantly different to XEP-0384 [16] in its approach both to what parts of a message are encrypted and to how key information is communicated.

5 Definition

5.1 Identities

XMPP identities (JIDs) shall be as defined in IETF RFC 7622 [10]. They are of the form *localpart@domainpart/resourcepart*. For the present document *localpart* is defined as a telephone number, in international format including the preceding + sign. This shall then be used, together with the current time, to create the month-augmented tel URI [3] used by ETSI TS 103 816-2 [12],thus allowing the same certificates - not just the same certificate format - to be used as by [12].

It would be advantageous if the *domainpart* and/or the *resourcepart* of the JID could be used to identify the cryptographic domain, and thus which certificate to use, but this is outside the scope of the present document. The present document supports any use of the *domainpart* and *resourcepart*, it constrains only the use of the *localpart* as described, and uses only that part.

5.2 XML Namespace

All new XML elements defined in the present document use an `xmlns` attribute to specify a namespace specific to the present document. Note that this includes cases where an element has the same name but a different format to a similar element defined in another specification, for example the element `<encrypted/>`. A namespace based on "*yourcompany.com*" domain name is suggested.

5.3 Provisioning

The provisioning of an endpoint is carried out by its KMS. This provides the endpoint, which has an identity agreed with the KMS, with its private key material and with a certificate for the domain it is in, and certificates for all domains that may include other endpoints with which it may communicate in either direction. The certificate format defined in ETSI TS 103 816-1 [11] shall be used. How the endpoint receives that private key material and certificates is outside the scope of the present document.

5.4 MIKEY-SAKKE Message

A first endpoint that has a message to send to a second endpoint shall construct a MIKEY-SAKKE message as defined in ETSI TS 103 816-2 [12] including all payloads there defined, including a unique RAND payload and a timestamp. The cryptographic identities of the endpoints required are determined from the JIDs of the endpoints as described in clause 5.1. To construct this message a SSV as defined in ETSI TS 103 816-2 [12] shall be used; this SSV shall be newly created for each message, shall not be revealed to any other parties except as encrypted in the MIKEY- SAKKE message, and is used to encrypt information in the XMPP message as described in clauses 5.5, 5.7 and 5.10.

This MIKEY-SAKKE message shall be base64 encoded as defined in IETF RFC 4648 [4] and included as the data in a new `<mikey/>` element within the `<body/>` element of the `<message/>` element, see clause 5.7.

The MIKEY-SAKKE message is authenticated by its inclusion of an Elliptic Curve-based Certificateless Signatures for IBE signature IETF RFC 6507 [7] (ECCSI) based on the source's cryptographic identity. It, and the message as a whole, is protected from a replay attack due to its included timestamp. A recipient shall reject a message that fails this authentication test or is unacceptably late based on its timestamp. The definition of "unacceptably late" is a matter for agreement between source and destination, or chosen by the system they both use on their behalf, and is outside the scope of the present document.

5.5 Symmetric Key

The SSV transferred using a MIKEY-SAKKE message is used at the source to create a symmetric key used for encryption of information that is included within that XMPP message and at the destination, after decryption, to recover that symmetric key and use it for decryption of that information.

For greater commonality with ETSI TS 103 816-2 [12], the Pseudo-Random Function (PRF) based process defined in IETF RFC 3830 [2] for converting the SSV to a symmetric encryption key shall also be used by the present document. However, as used by [12] this PRF-derived key is not the key used for symmetric encryption, but is the master key passed to SRTP, which in turn derives a session key from it. That subsequent step is not included in the present document. In particular issues that may arise for a long SRTP session that indicate the need for rekeying are not an issue for the present document where only a single message, plus possibly a short message receipt, is encrypted. A final key created as defined in [12], following [2], is the key used by the present document as a message encryption key.

The PRF-based process defined in IETF RFC 3830 [2] requires a constant based on the function for which the derived key is to be used, and the constant **0x2AD01C64** that defines the traffic encryption key Traffic Encryption Key (TEK) in [2] shall be used by the present document.

5.6 XMPP Message Format

The present document primarily considers the message stanza in an XMPP message, included as the XML element `<message/>`. A message encrypted using the present document shall include a message stanza. This stanza shall have five common attributes `to`, `from`, `id`, `type` and `xml:lang` that are specified in clause 8.1 of IETF RFC 6120 [5]. The former two carry endpoint identities (JIDs) and shall be used.

The attribute `id` is not affected by the present document, it is a locally generated sufficiently unique identity used to associate responses, including message receipts and errors, to the stanza. The attribute `id` shall therefore be sufficiently unique for this purpose - i.e. it cannot be reused for the same destination within any period in which a message receipt or error could be received; stronger uniqueness than this should be used, see clause 8.1.3 of IETF RFC 6120 [5], however the minimum requirement there of uniqueness within a stream is not sufficient for the present document as message sent using the present document do not share a stream.

Possible values of the attribute `type` for messages are specified in clause 5.2.2 of IETF RFC 6121 [6]; this would suggest that the appropriate value for messages sent using the present document would be `normal`, but for greater compatibility with existing XMPP implementations the value `chat` shall be used. The attribute `xml:lang` specifies the default language of any character data that is intended to be presented to a human user; this is also not affected by the present document.

5.7 Message Encryption

Like the voice data encryption used by ETSI TS 103 816-2 [12], the message encryption used by the present document uses AES-GCM, an Authenticated Encryption with Associated Data (AEAD) algorithm. This algorithm is also used by XEP-0384 [16], and thus some of the latter's approach to message stanza encryption is followed.

The message encryption steps which shall be used are:

- The source creates the XMPP message to be encrypted.
- The source creates a 16-octet SSV that is unique to this message.
- The source creates a MIKEY-SAKKE message using that SSV as defined in clause 5.4.
- The source creates an encryption key from the SSV as defined in clause 5.5.
- The source creates an initialization vector (IV) as defined in ETSI TS 103 816-2 [12]; details of the means of creation are not required to be included in the message, instead as indicated the full IV is included in the message.

- The source encrypts the concatenation (there may be whitespace between elements) of the <message/>, <presence/> and <iq/>elements (the latter two of which may be used) using that encryption key as defined in ETSI TS 103 816-2 [12]. After this step, the encryption key shall only be used if requesting a message receipt, see clause 5.9; the encryption key shall be erased as soon as possible, this should be immediately after this step if a message receipt is not requested.

All options supported in ETSI TS 103 816-2 [12] may be used, in particular the use of either the 128- or 256-bit versions of AES- GCM; the former should be used.

There are now three required pieces of binary data: the MIKEY-SAKKE message, the initialization vector and the message ciphertext. These are all base64 encoded, as defined in IETF RFC 4648 [4], so that they can be carried in XML elements as text and without having to handle XML whitespace issues. Note that the use of IETF RFC 4648 [4] for base64 encoding replaces the use of IETF RFC 2045 [1] for that purpose in XEP-0384 [16].

To pass that information to the receiving endpoint, a new <message/> element (stanza) shall be created as follows, adapted from XEP-0384 [16]. This single element shall replace all three unencrypted stanzas. Note that the <header/> and <encrypted/>elements are both new as defined in clause 5.2.

- The new <message/> element has the same attributes as the unencrypted <message/> element.
- The new <message/> element contains only one element, a <body/> element with no attributes. Within this <body/> element is included a <header/> element and an <encrypted/> element. These two elements shall both have an xmlns attribute specifying the namespace defined in clause 5.2.
- The <header/> element shall include a versionattribute; to be compliant with this version of the present document that attribute shall have the value **1.0**. Such a <header/> element shall contain a <mikey/> element as defined in clause 5.4. Note that this element replaces the <key/> element in XEP-0384 [16].
- The <encrypted/> element has an attribute algorithm that shall be one of the strings **aes128-gcm** or **aes256-gcm**, according to the choice made in clause 4.7. The <encrypted/> element contains two elements:
  - An <iv/> element, as in XEP-0384 [16], that is used to contain the encryption initialization vector (IV) used. The latter shall be base64 encoded as defined in IETF RFC 4648 [4]. This is the only content of this element.
  - A <data/> element that contains the encrypted message after base64 encoding as defined in IETF RFC 4648 [4]. This is the only content of this element.

5.8 Message Decryption

A received message shall be decrypted and authenticated at its destination by:

- Recovering the three pieces of binary data (the MIKEY-SAKKE message, the IV and the message ciphertext) by extracting the three base64 encoded fields and reversing the base64 encoding. Message decryption fails if all three cannot be extracted.
- Parsing the MIKEY-SAKKE message and authenticating it using its included ECCSI signature. Message decryption fails if either parsing or authentication fails.
- Extracting the encrypted SSV from the MIKEY-SAKKE message, decrypting it and forming a decryption key as described in ETSI TS 103 816-2 [12]. Message decryption fails if this process fails.
- Using the decryption key and IV to decrypt the message ciphertext. This process fails if the decryption key or IV are incorrect (in particular if either the key or IV is the wrong size) or if the decryption authentication fails.
- Verifying that the decrypted message consists of a valid <message/> element (shall be done) and <presence/> and/or <iq/> elements (may be done). Decryption fails if this is not so. Note that this does not require all elements within these three elements to be recognized. Unrecognized elements within these elements shall be ignored provided that they are suitably (unknown) namespace qualified.

- Verifying that the attributes of the decrypted <message/> element match those of the received <message/> element. Decryption fails if this is not so.

If message decryption fails for any of the indicated reasons, then the received message shall be rejected.

5.9 Message Delivery Receipt

An endpoint that is compliant with the present document shall implement message delivery receipt as defined in XEP-0184 [14].

Message delivery receipt is requested by the inclusion of a <request/> element within the message stanza. If requested, then this shall be added to the unencrypted message stanza before encryption as defined in clause 5.7.

Message delivery receipts are XMPP messages containing a <received/> element (only) within the <message/> element. Following clause 8.1.3 of IETF RFC 6120 [5], the message delivery receipt shall have the same id attribute as the acknowledged message. Following the requirement for this attribute in clause 5.6 there will be no confusion as to the matching of messages and receipts if the latter are received out of order, but within an expected time period.

Message delivery receipts shall be encrypted as described in clause 5.7 except that no MIKEY- SAKKE message is required, and thus no <header/> element is required. Instead the same encryption algorithm and key is used as for the original message encryption. The IV used shall be different from that used for the received message, and for all counter values. This may be implemented by ensuring that the salt used as defined in ETSI TS 103 816-2 [12] is different.

Encryption of the message receipt with the same key requires that a sender that has requested message delivery receipt shall retain the encryption key used until it receives a receipt, after which it shall delete that key. However, as a receipt may not be received, the sender shall also delete the encryption key immediately after the expiry of the expected time within which a message receipt is expected; this time shall not be unnecessarily long.

An endpoint compliant with the present document should either request message delivery receipt or provide a means by which a user can select or reject the sending of message delivery receipt requests. Such an endpoint should also either acknowledge a message delivery receipt request or provide a means by which a user can select or reject the sending of message delivery receipt request acknowledgements (message delivery receipts).

Acknowledgement of the receipt of a message that is rejected for any reason, including due to decryption, including authentication, failure, is a tradeoff between a security preference for not acknowledging the receipt of what might be an unauthorized message, and a performance issue where a rejection might be due to a temporary problem and re-sending the message would improve that. The decision as to which approach to follow is beyond the scope of the present document, which permits either behaviour.

Note that the limitations on message receipt request described in XEP-0184 [14] apply, in particular that an acknowledgement is only sent on receipt and that a message delivery receipt provides no evidence that a message has been read by any user. However, the authentication process used by the present document indicates that the message did reach an endpoint with the expected security credentials.

5.10 Attached Content

For the purposes of the present document, an attachment to a message is a file that is encrypted and stored on a server that is accessible by the message recipient. A message may "contain" (i.e. provide a link to) one or more attachments.

Such a file shall be encrypted using AES-GCM; AES-128 should be used, but AES-256 shall also be supported. Note that this is an authenticated encryption algorithm. The file stored at the URL is simply that encrypted file with no further information provided on the server. Two additional pieces of information are required to decrypt the file: the file encryption key and the Initialization Vector (IV). The former shall not be stored at a publicly accessible location, the latter could be so stored, but need not be, and using the present document is not. Because files may be shared with different combinations of recipients, each file shall have its own key, generated randomly for it; the IV used shall also be generated specifically for that file as described in ETSI TS 103 816-2 [12]; any salting option described therein may be used when using the present document; only the final IV is used by the present document.

Thus, in order to provide the attachment to the recipient the following shall be provided: the URL, the encryption algorithm, which will be indicated by one of the strings **aes128-gcm** or **aes256-gcm**, the IV and the encryption key. Other file information may be provided and used as described below.

Each attached file is defined within the unencrypted message stanza, using an XML element `<content/>`. This name, and some details of it are taken from XEP-0234 [15], but this is a new element as defined in clause 5.2 and thus shall have an appropriate `xmlns` attribute. Each attached file has its own `<content/>` element. Note that the protocol aspects of XEP-0234 [15], known as Jingle, are NOT used by the present document.

The attached file is specified within the `<content/>` element using a `<description/>` element, the name again taken from XEP-0234 [15]. XEP-0234 [15] then allows details of the file itself to be provided using a `<file/>` element within the `<description/>` element. That `<file/>` element is also used by the present document. All elements within the `<file/>` element, and the `<file/>` element itself, may be used. (Most of these elements are also optional in XEP-0234 [15]). Note that of those elements, the file's hash is not needed to authenticate the file due to the authenticated encryption of the file. The file's hash could be used to avoid an unnecessary download of an already available file; however, this use is beyond the scope of the present document. Note also that because the unencrypted file's type may be inappropriate for the encrypted file, the `<name/>` element that provides that possibly otherwise lost information is expected to be commonly used.

The information specific to the present document is included in an element `<reference/>` in the `<content/>` element. That information is:

- The URL, using element `<url/>`.
- The file encryption information, using an element `<encryption/>`. The encryption algorithm used is an attribute `algorithm` of this element that shall be one of the strings **aes128-gcm** or **aes256- gcm**. Within this element are:
  - The file key, using element `<key/>`. The key shall be base64 encoded as defined in IETF RFC 4648 [4]. Note that at this point this key is unencrypted.
  - The initialization vector, using element `<iv/>`. This should be created as indicated in clause 5.7, but as long as the full IV is made available the details of that creation are not required by the message recipient. The initialization vector shall be base64 encoded as defined in IETF RFC 4648 [4].

The `<content/>` element(s) shall be included directly in the `<message/>` element before using the encryption process defined in clause 4.7. Note that this ensures that the attached file key is encrypted before being sent. Because the SSV and IV are uniquely created for this message, the attached file key is encrypted using key material that is never reused.

Annex A (informative): Examples

A.1 Example Message

From clause 5.2.4 of IETF RFC 6121 [6], an example message stanza using a `<message/>` element is:

```
<message
    from='juliet@example.com/balcony'
    id='c8xg3nf8'
    to='romeo@example.net'
    type='chat'
    xml:lang='en'>
    <subject>I implore you!</subject>
    <body>Wherefore art thou, Romeo?</body>
</message>
```

Note that to correspond to Request For Comments (usually an IETF document, often a standard; the term request for comments is historical) (RFC) rules on creating examples, this uses the two domains `example.com` and `example.net` rather than the more obvious `capulet.com` and `montague.com` (which might be in real use). This practice is continued in this example. Both JIDs here have a *domainpart* and Juliet's JID also has a *resourcepart*, these are retained in this example to illustrate that the present document can handle these JID parts.

However, as defined in clause 5.1, when using the present document the endpoints are associated not with a name (`juliet`, `romeo`) but with a telephone number. The phone numbers used here will be `+447700585438` and `+447700766386`. Thus, Juliet will have a JID of `+447700585438@example.com/balcony` and Romeo will have a JID of `+447700766386@example.net`

The unencrypted message used in this example will thus be:

```
<message
      from='+447700585438@example.com/balcony'
      id='c8xg3nf8'
      to='+447700766386@example.net'
      type='chat'
      xml:lang='en'>
      <subject>I implore you!</subject>
      <body>Wherefore art thou, Romeo?</body>
</message>
```

For the month of July 1591, Juliet has a cryptographic identity of `1591-07\0tel:+447700585438\0` and Romeo has a cryptographic identity of `1591-07\0tel:+447700766386\0`. Juliet is provisioned by her KMS with private key material for her cryptographic identity, and Romeo is provisioned by his KMS with private key material for his cryptographic identity. Both are provided with their own and each other's domain certificates. They are assumed to know which of their provisioned certificates is that of the other's domain.

Juliet now creates a MIKEY-SAKKE message intended for Romeo, as described in [9]. Doing so includes creating a 16 octet SSV that she will use further as well as using it to create the MIKEY-SAKKE message. She then base64 encodes the MIKEY-SAKKE message as described in IETF RFC 4648 [4]. The result of that process will be represented as *base64 encoded MIKEY-SAKKE message* in figures.

Juliet then forms a 16-octet encryption key from the 16-octet SSV using the algorithm described in clause 4.1 of IETF RFC 3830 [2], specifically as to form a TEK using the constant **0x2AD01C64**. She uses that key to encrypt the complete `<message/>` element. Note that however Juliet chooses to handle white space, that in this `<message/>` element will simply be replicated when Romeo decrypts the message.

Juliet encrypts the message using the 128-bit version of AES-GCM as described in [12]. As part of that process she creates a 16-octet initialization vector (IV). She base64 encodes that IV as described in IETF RFC 4648 [4]. The result of that process will be represented as *base64 encoded IV* in figures. The result of the encryption of the `<message/>` element is also base64 encoded and is represented in figures as *base64 encoded encrypted message element*.

These three items of base64 encoded data are put into the message within a `<body/>` element as described in clause 5.7. The namespace assumed for this example is `https://.mantua.org/xmpp`. This produces the following encrypted element:

```
<message
   from='+447700585438@example.com/balcony'
   id='c8xg3nf8'
   to='+447700766386@example.net'
   type='chat'
   xml:lang='en'>
   <body>
    <header xmlns='https://www.mantua.org/xmpp' version='1.0'>
      <mikey>
        base64 encoded MIKEY-SAKKE message
      </mikey>
    </header>
    <encrypted algorithm='aes128-gcm'>
      <iv>
        base64 encoded IV
      </iv>
      <data>
        base64 encoded encrypted message element
      </data>
    </encrypted>
   </body>
</message>
```

Note that when Romeo receives this message, when he decrypts the message element, he will get a second copy of the attributes `from`etc. He will reject the message if these do not match the received unencrypted attributes.

If Juliet adds a `<presence/>` and an `</iq>` stanza to form the message, the result is:

```
<message
   from='+447700585438@example.com/balcony'
   id='c8xg3nf8'
   to='+447700766386@example.net'
   type='chat'
   xml:lang='en'>
   <body>
    <header xmlns='https://www.mantua.org/xmpp' version='1.0'>
      <mikey>
        base64 encoded MIKEY-SAKKE message
      </mikey>
    </header>
    <encrypted algorithm='aes128-gcm'>
      <iv>
        base64 encoded IV
      </iv>
      <data>
        base64 encoded encrypted message element
      </data>
    </encrypted>
   </body>
</message>
<presence>
    presence information
</presence>
<iq/>
    IQ information
</iq>
```

She will then encrypt that message from the opening `<` of the `<message/>` element to the closing `>` of the `<iq/>` element. Her message to Romeo will have the same format as previously, without either a `<presence/>` or an `<iq/>` element.

A.2 Example Message with Attachment

It is now assumed that Juliet wishes to send a link to the file `soliloquy.pdf` to Romeo. She arranges for it to be encrypted using the 128-bit version of AES-GCM, and stored at the publicly accessible URL `https://www.example.com/balcony/soliloquy.txt`. The key used to encrypt the file is represented in figures as *base64 encoded attachment key*. The initialization vector used to encrypt the file is also base64 encoded and represented in figures as *base64 encoded attachment IV*.

The unsigned message from clause A.1, in this example without presence or IQ information, now has the attachment added. Juliet chooses to send only the file name as information about the file. This then produces the unencrypted message:

```
<message
   from='+447700585438@example.com/balcony'
   id='c8xg3nf8'
   to='+447700766386@example.net'
   type='chat'
   xml:lang='en'>
   <subject>I implore you!</subject>
   <body>Wherefore art thou, Romeo?</body>
   <content xmlns='https://www.mantua.org/xmpp'>
      <description>
         <file>
            <name>soliloquy.pdf</name>
         </file
      </description>
      <reference>
         <url>https://www.example.com/balcony/soliloquy.txt</url>
         <encryption algorithm='aes128-gcm'>
            <key>
               base64 encoded attachment key
            </key>
            <iv>
               base64 encoded attachment IV
            </iv>
         </encryption>
      </reference>
   </content>
</message>
```

This message will then be encrypted, appearing to any other party equivalent to a message without an attachment, with exactly the same format as shown in clause A.1. Note that messages with and without attachments but otherwise identical will have different encrypted lengths, but so also will messages with different text message lengths (subject and body length).

A.3 Example Message Receipt Request

If Juliet wishes her message (in this example without attachments or presence or IQ information) to be acknowledged, then her unencrypted message becomes:

```
<message
      from='+447700585438@example.com/balcony'
      id='c8xg3nf8'
      to='+447700766386@example.net'
      type='chat'
      xml:lang='en'>
      <body>
            <header xmlns='https://www.mantua.org/xmpp' version='1.0'>
               <mikey>
                     base64 encoded MIKEY-SAKKE message
               </mikey>
            </header>
            <encrypted algorithm='aes128-gcm'>
               <iv>
                     base64 encoded IV
               </iv>
               <data>
                     base64 encoded encrypted message element
               </data>
            </encrypted>
```

```
      </body>
      <request xmlns='urn:xmpp:receipts'/>
</message>
```

Romeo will now send the following unencrypted message receipt. Note that it has the same `id` attribute as the message that it acknowledges.

```
<message
      from=' +447700766386@example.net'
      id='c8xg3nf8'
      to='+447700585438@example.com/balcony'
      type='chat'
      xml:lang='en'>
      <received xmlns='urn:xmpp:receipts' id=c8xg3nf8'/>
</message>
```

Which on encryption becomes:

```
<message
      from='+447700766386@example.net'
      id='c8xg3nf8'
      to='+447700585438@example.com/balcony'
      type='chat'
      xml:lang='en'>
      <body>
          <encrypted algorithm='aes128-gcm'>
              <iv>
                  base64 encoded IV
              </iv>
              <data>
                  base64 encoded encrypted message element
              </data>
          </encrypted>
      </body>
</message>
```

Note that the two base64 encoded fields are different to those in Juliet's message.

History

Document history		
V1.1.1	July 2021	Publication