Internet Engineering Task Force (IETF)
Request for Comments: 6267
Category: Informational
ISSN: 2070-1721

V. Cakulev
G. Sundaram
Alcatel Lucent

June 2011

MIKEY-IBAKE: Identity-Based Authenticated Key Exchange (IBAKE) Mode of Key Distribution in Multimedia Internet KEYing (MIKEY)

Abstract

This document describes a key management protocol variant for the Multimedia Internet KEYing (MIKEY) protocol that relies on a trusted key management service. In particular, this variant utilizes Identity-Based Authenticated Key Exchange (IBAKE) framework that allows the participating clients to perform mutual authentication and derive a session key in an asymmetric Identity-Based Encryption (IBE) framework. This protocol, in addition to providing mutual authentication, eliminates the key escrow problem that is common in standard IBE and provides perfect forward and backward secrecy.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6267.

Table of Contents

1. Introduction 3
2. Terminology 4
2.1. Requirements Language 4
2.2. Definitions and Notation 4
2.3. Abbreviations 5
3. Use Case Scenarios 6
3.1. Forking 6
3.2. Retargeting 6
3.3. Deferred Delivery 7
4. MIKEY-IBAKE Protocol Description 7
4.1. Overview 7
4.2. Message Exchanges and Processing 10
4.2.1. REQUEST_KEY_INIT/REQUEST_KEY_RESP Message Exchange . . 10
4.2.2. I_MESSAGE/R_MESSAGE Message Exchanges 12
5. Key Management 16
5.1. Generating Keys from the Session Key 17
5.2. Generating Keys for MIKEY Messages 17
5.3. CSB Update 18
5.4. Generating MAC and Verification Message 18
6. Payload Encoding 19
6.1. Common Header Payload (HDR) 19
6.1.1. IBAKE Payload 20
6.1.2. Encrypted Secret Key (ESK) Payload 21
6.1.3. Key Data Sub-Payload 21
6.1.4. EC Diffie-Hellman Sub-Payload 22
6.1.5. Secret Key Sub-Payload 23
7. Security Considerations 24
7.1. General Security Considerations 24
7.2. IBAKE Protocol Security Considerations 25
7.3. Forking 26
7.4. Retargeting 26
7.5. Deferred Delivery 26
8. IANA Considerations 27
9. References 28
9.1. Normative References 28
9.2. Informative References 29

1. Introduction

The Multimedia Internet Keying (MIKEY) [RFC3830] specification describes several modes of key distribution solution that address multimedia scenarios using pre-shared keys, Public Keys, and optionally a Diffie-Hellman key exchange. Multiple extensions of MIKEY have been specified, such as HMAC-Authenticated (Hashed Message Authentication Code) Diffie-Hellman [RFC4650] and MIKEY-RSA-R [RFC4738].

To address deployment scenarios in which security systems serve a large number of users, a key management service is often preferred. With such a service in place, it would be possible for a user to request credentials for any other user when they are needed. Some proposed solutions [RFC6043] rely on Key Management Services (KMSs) in the network that create, distribute, and manage keys in a real time. Due to this broad functionality, key management services would have to be online, maintain high availability, and be networked across operator boundaries.

This document describes a solution in which KMSs are low-availability servers that communicate with end-user clients periodically (e.g., once a month). The online transactions between the end-user clients (for media plane security) are based on Identity-Based Encryption (IBE) [BF]. These online transactions between the end-user clients allow them to perform mutual authentication and derive a session key not known to any external entity (including KMSs). This protocol, in addition to providing keys not known to any external entity and allowing for end-user clients to mutually authenticate each other (at the media plane layer), provides perfect forward and backward secrecy. In this protocol, the KMS-to-client exchange is used sparingly (e.g., once a month); hence, the KMS is no longer required to be a high-availability server, and in particular different KMSs don’t have to communicate with each other (across operator boundaries). Moreover, given that an IBE is used, the need for costly Public Key Infrastructure (PKI) and all the operational costs of certificate management and revocation are eliminated. This is achieved by concatenating Public Keys with a date field, thereby ensuring corresponding Private Keys change with the date and, more importantly, limiting the damage due to loss of a Private Key to just that date while not requiring endpoints involved in communication to be time synchronized. The granularity in the date field is a matter of security policy and deployment scenario. For instance, an operator may choose to use one key per day and hence the KMS may issue Private Keys for a whole subscription cycle at the beginning of a subscription cycle. Therefore, unlike in the PKI systems, where issued certificate is typically valid for period of time thereby requiring revocation procedures to limit their validity, the scheme

described in this document uses time-bound public identities, which automatically expire at the end of a time span indicated in the identity itself. With the self-expiration of the public identities, the traditional real-time validity verification and revocation is not required. For example, if the public identity is bound to one day, then, at the end of the day, the Public/Private Key pair issued to this peer will simply not be valid anymore. Nevertheless, just like with Public-Key-based certificate systems, if there is a need to revoke keys before the designated expiry time, communication with a third party will be needed.

Additionally, various call scenarios are securely supported -- this includes secure forking, retargeting, deferred delivery and pre-encoded content.

MIKEY is widely used in the 3GPP community. This specification is intended primarily for use with 3GPP media security, but it may also be applicable in Internet applications.

2. Terminology

2.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2.2. Definitions and Notation

IBE Encryption: Identity-Based Encryption (IBE) is a Public-Key encryption technology that allows a Public Key to be calculated from an identity, and the corresponding Private Key to be calculated from the Public Key. [RFC5091], [RFC5408], and [RFC5409] describe algorithms required to implement the IBE.

(Media) session: The communication session intended to be secured by the MIKEY-IBAKE provided key(s).

E(k, x)	Encryption of x with the key k
[x]P	Point multiplication on an elliptic curve, i.e., adding a point P to itself total of x times
K_PUBx	Public Key of x
[x]	x is optional
{x}	Zero or more occurrences of x
(x)	One or more occurrences of x
||	Concatenation
|	OR (selection operator)

2.3. Abbreviations

EC	Elliptic Curve
ESK	Encrypted Secret Key
HMAC	Hashed Message Authentication Code
IBE	Identity-Based Encryption
I	Initiator
IBAKE	Identity-Based Authenticated Key Exchange
IDRi	Initiator’s Identity
IDRr	Responder’s Identity
KMS	Key Management Service
K_PR	Private Key
K_PUB	Public Key
K_SESSION	Session Key
MAC	Message Authentication Code
MIKEY	Multimedia Internet KEYing
MKI	Master Key Identifier
MPK	MIKEY Protection Key
PKI	Public Key Infrastructure
PRF	Pseudorandom Function
R	Responder
SK	Secret Key
SIP	Session Initiation Protocol
SPI	Security Parameter Index

SRTP Secure Realtime Transport Protocol

TEK Traffic Encryption Key

TGK TEK Generation Key

3. Use Case Scenarios

This section describes some of the use case scenarios supported by MIKEY-IBAKE, in addition to regular two-party communication.

3.1. Forking

Forking is the delivery of a request (e.g., SIP INVITE message) to multiple endpoints. This happens when a single user is registered more than once. An example of forking is when a user has a desk phone, PC client, and mobile handset all registered with the same public identity.

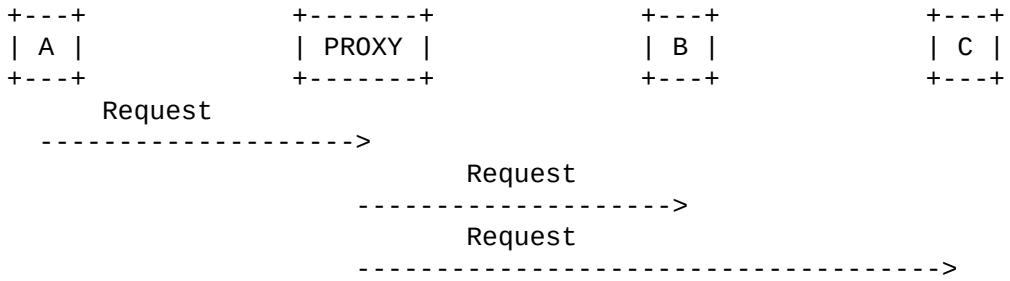


Figure 1: Forking

3.2. Retargeting

Retargeting is a scenario in which a functional element decides to redirect the session to a different destination. This decision to redirect a session may be made for different reasons by a number of different functional elements and at different points in the establishment of the session.

There are two basic scenarios of session redirection. In scenario one, a functional element (e.g., Proxy) decides to redirect the session by passing the new destination information to the originator. As a result, the originator initiates a new session to the redirected destination provided by the Proxy. For the case of MIKEY-IBAKE, this means that the originator will initiate a new session with the identity of the redirected destination. This scenario is depicted in Figure 2 below.

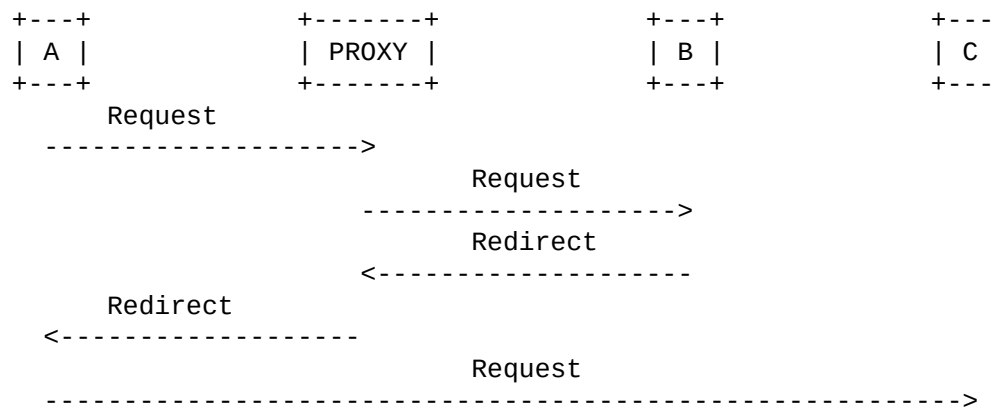


Figure 2: Retargeting

In the second scenario, a proxy decides to redirect the session without informing the originator. This is a common scenario specified in SIP [RFC3261].

3.3. Deferred Delivery

Deferred delivery is a type of service such that the session content cannot be delivered to the destination at the time that it is being sent (e.g., the destination user is not currently online). Nevertheless, the sender expects the network to deliver the message as soon as the recipient becomes available. A typical example of deferred delivery is voicemail.

4. MIKEY-IBAKE Protocol Description

4.1. Overview

Most of the previously defined MIKEY modes consist of a single (or half) roundtrip between two peers. MIKEY-IBAKE consists of up to three roundtrips. In the first roundtrip, users (Initiator and Responder) obtain their Private Key(s) (K_PR) from the KMS. This roundtrip can be performed at anytime and, as explained earlier, takes place, for example, once a month (or once per subscription cycle). The second and the third roundtrips are between the Initiator and the Responder. Observe that the Key Management Service is only involved in the first roundtrip. In Figure 3, a conceptual signaling diagram for the MIKEY-IBAKE mode is depicted.

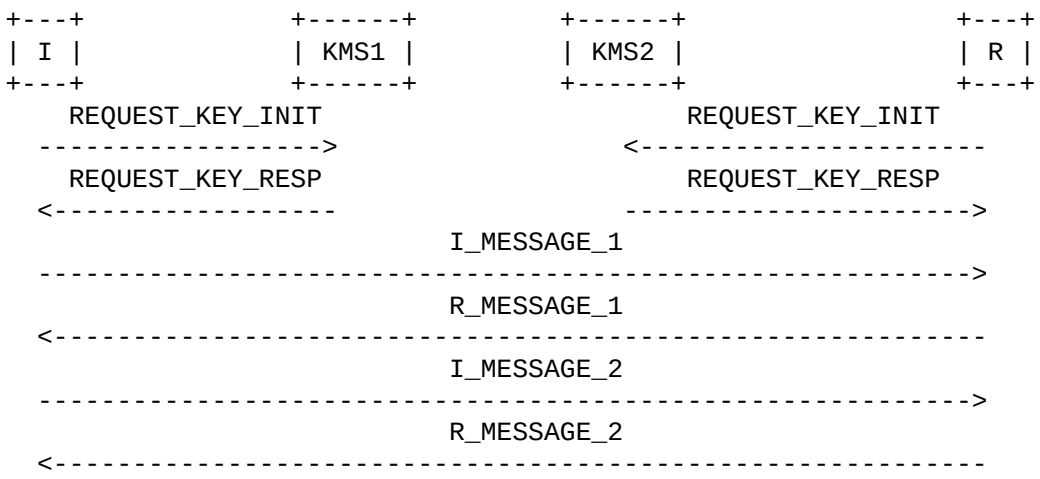


Figure 3: Example Message Exchange

The Initiator (I) wants to establish a secure media session with the Responder (R). The Initiator and the Responder trust a third party, the Key Management Service (KMS), with which they both have, or can establish, shared credentials. These pre-established trust relations are used by a user (i.e., Initiator and Responder) to obtain Private Keys. Rather than a single KMS, several different KMSs may be involved, e.g., one for the Initiator and one for the Responder as shown in Figure 3. The Initiator and the Responder do not share any credentials; however, the Initiator knows the Responder's public identity. The assumed trust model is illustrated in Figure 4.

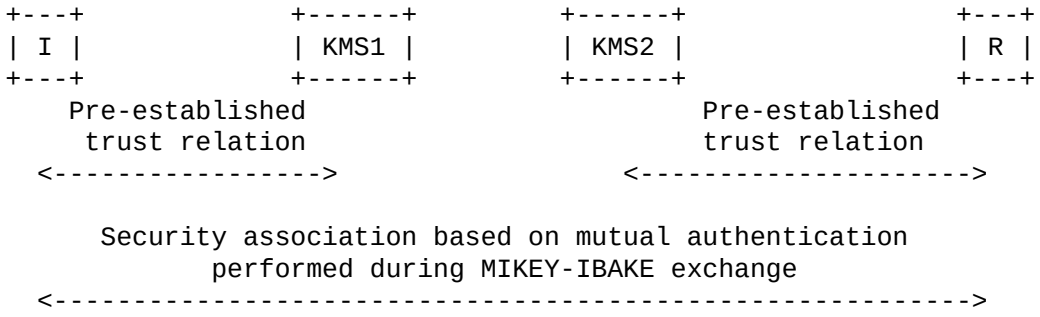


Figure 4: Trust Model

Below, a description of how Private Keys are obtained using MIKEY messages is provided. An alternative way for obtaining Private Keys using HTTP is described in [RFC5408].

The Initiator obtains Private Key(s) from the KMS by sending a REQUEST_KEY_INIT message. The REQUEST_KEY_INIT message includes Initiator's public identity(s) (if the Initiator has more than one public identity, it may request Private Keys for every identity registered) and is protected via a MAC based on a pre-shared key or via a signature (similar to the MIKEY-PSK and MIKEY-RSA modes). If the request is authorized, the KMS generates the requested keys, encodes them, and returns them in a REQUEST_KEY_RESP message. This exchange takes place periodically and does not need to be performed every time an Initiator needs to establish a secure connection with a Responder.

The Initiator next chooses a random x and computes [x]P, where P is a point on elliptic curve E known to all users. The Initiator uses the Responder's public identity to generate the Responder's Public Key (e.g., K_PUBr=H1(IDRr||date)), where Hi is hash function known to all users, and the granularity in date is a matter of security policy and known publicly. Then the Initiator uses this generated Public Key to encrypt [x]P, IDRi and IDRr and includes this encrypted information in an I_MESSAGE_1 message, which is sent to the Responder. The encryption is Identity-Based Encryption (IBE) as specified in [RFC5091] and [RFC5408]. In turn, the Responder IBE-decrypts the received message using its Private Key for that date, chooses random y and computes [y]P. Next, the Responder uses Initiator's identity obtained from I_MESSAGE_1 to generate Initiator's Public Key (e.g., K_PUBi=H1(IDRi||date)) and IBE-encrypts (IDRi, IDRr, [x]P, [y]P) using K_PUBi, and includes it in R_MESSAGE_1 message sent to the Initiator. At this point, the Responder is able to generate the session key as [x][y]P. This session key is then used to generate TGK as specified in Section 5.1.

Upon receiving and IBE-decrypting an R_MESSAGE_1 message, the Initiator verifies the received [x]P. At this point, the Initiator is able to generate the same session key as [x][y]P. Upon successful verification, the Initiator sends I_MESSAGE_2 message to the Responder, including IBE-encrypted IDRi, IDRr and previously received [y]P. The Responder sends a R_MESSAGE_2 message to the Initiator as verification.

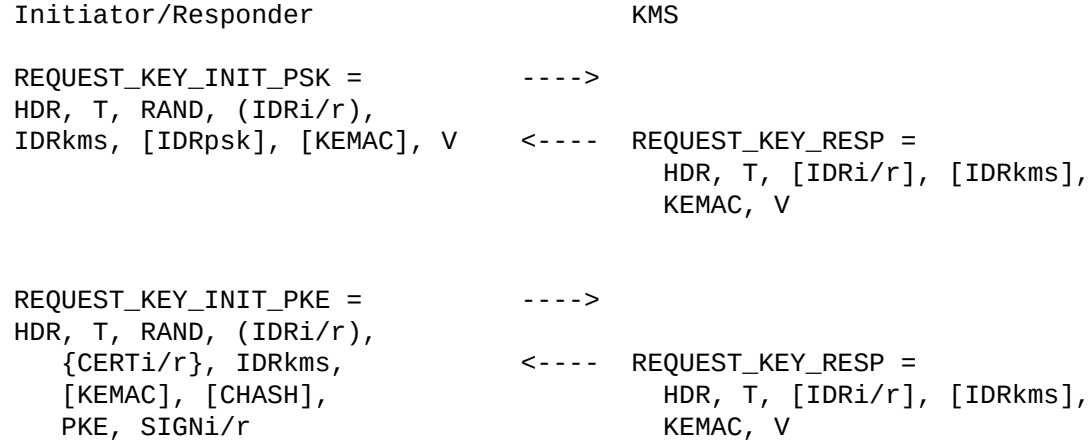
The above described is the most typical use case; in Section 3, some alternative use cases are discussed.

MIKEY-IBAKE is based on [RFC3830]; therefore, the same terminology, processing, and considerations still apply unless otherwise stated. Payloads containing EC Diffie-Hellman values and keys exchanged in I_MESSAGE/R_MESSAGE are IBE encrypted as specified in [RFC5091] and [RFC5408], while the keys exchanged in KEY_REQUES_INIT/ KEY_REQUEST_RESPONSE are encrypted as specified in [RFC3830]. In all exchanges, encryption is only applied to the payloads containing keys and EC Diffie-Hellman values and not to the entire messages.

4.2. Message Exchanges and Processing

4.2.1. REQUEST_KEY_INIT/REQUEST_KEY_RESP Message Exchange

This exchange is used by a user (e.g., Initiator or Responder) to request Private Keys from a trusted Key Management Service, with which the user has pre-shared credentials. A full roundtrip is required for a user to receive keys. As this message must ensure the identity of the user to the KMS, it is protected via a MAC based on a pre-shared key or via a signature. The initiation message REQUEST_KEY_INIT comes in two variants corresponding to the pre-shared key (PSK) and Public-Key encryption (PKE) methods of [RFC3830]. The response message REQUEST_KEY_RESP is the same for the two variants and SHALL be protected by using the pre-shared/envelope key indicated in the REQUEST_KEY_INIT message.



4.2.1.1. Components of the REQUEST_KEY_INIT Message

The main objective of the REQUEST_KEY_INIT message is for a user to request one or more Private Keys (K_PR) from the KMS. The user may request a K_PR for each public identity it possesses, as well as for multiple dates.

The REQUEST_KEY_INIT message MUST always include the Header (HDR), Timestamp (T), and RAND payloads. The CSB ID (Crypto Session Bundle ID) SHALL be assigned as in [RFC3830]. The user SHALL include it in the CSB ID field of the Header. The user SHALL set the #CS field to '0' since CS (Crypto Session(s)) SHALL NOT be handled. The CS ID map type SHALL be the "Empty map" as defined in [RFC4563].

IDRi/r contains the identity of the user. Since the user may have multiple identities, multiple IDRi/r fields may appear in the message.

IDRkms SHALL be included.

The KEMAC payload SHALL be used only when the user needs to use specific keys. Otherwise, this payload SHALL NOT be used.

4.2.1.1.1. Components of the REQUEST_KEY_INIT_PSK Message

The IDRpsk payload MAY be used to indicate the pre-shared key used.

The last payload SHALL be a Verification (V) payload where the authentication key (auth_key) is derived from the pre-shared key (see Section 4.1.4 of [RFC3830] for key derivation specification).

4.2.1.1.2. Components of the REQUEST_KEY_INIT_PKE Message

The certificate (CERT) payload SHOULD be included. If a certificate chain is to be provided, each certificate in the chain MUST be included in a separate CERT payload.

The PKE payload contains the encrypted envelope key: PKE = E(PKkms, env_key). It is encrypted using the KMS's Public Key (PKkms). If the KMS possesses several Public Keys, the user can indicate the key used in the CHASH payload.

SIGNi/r is a signature covering the entire MIKEY message, using the Initiator's signature key.

4.2.1.2. Processing of the REQUEST_KEY_INIT Message

If the KMS can verify the integrity of the received message and the message can be correctly parsed, the KMS MUST check the Initiator's authorization. If the Initiator is authorized to receive the requested Private Key(s), the KMS MUST send a REQUEST_KEY_RESP message. Unexpected payloads in the REQUEST_KEY_INIT message SHOULD be ignored. Errors are handled as described in [RFC3830].

4.2.1.3. Components of the REQUEST_KEY_RESP Message

The version, PRF func and CSB ID, #CS, and CS ID map type fields in the HDR payload SHALL be identical to the corresponding fields in the REQUEST_KEY_INIT message. The KMS SHALL set the V flag to 0 and the user receiving it SHALL ignore it as it has no meaning in this context.

The Timestamp type and value SHALL be identical to the one used in the REQUEST_KEY_INIT message.

KEMAC = E(encr_key, (ID || K_PR))

The KEMAC payload SHOULD use the NULL authentication algorithm, as a MAC is included in the V payload. Depending on the type of REQUEST_KEY_INIT message, either the pre-shared key or the envelope key SHALL be used to derive the encr_key.

The last payload SHALL be a Verification (V) payload. Depending on the type of REQUEST_KEY_INIT message, either the pre-shared key or the envelope key SHALL be used to derive the auth_key.

4.2.1.4. Processing of the REQUEST_KEY_RESP Message

If the Initiator/Responder can correctly parse the received message, the received session information SHOULD be stored. Otherwise, the Initiator/Responder SHOULD silently discard the message and abort the protocol.

4.2.2. I_MESSAGE/R_MESSAGE Message Exchanges

This exchange is used for Initiator and Responder to mutually authenticate each other and to exchange EC Diffie-Hellman values used to generate TGK. These exchanges are modeled after the pre-shared key mode, with the exception that the Elliptic Curve Diffie-Hellman values and Secret Keys (SKs) are encoded in IBAKE and ESK payloads instead of a KEMAC payload. Two full roundtrips are required for this exchange to successfully complete. The messages are preferably included in the session setup signaling (e.g., SIP INVITE).

```
Initiator                                Responder

   I_MESSAGE_1 =                  ---->
   HDR, T, RAND, IDRi, IDRr,
      IBAKE, [ESK]                <----  R_MESSAGE_1 =
                                           HDR, T, IDRi,
                                           IDRr, IBAKE

   I_MESSAGE_2 =                  ---->
   HDR, T, RAND, IDRi, IDRr,
      IBAKE, [ESK]                <----  R_MESSAGE_2 =
                                         HDR, T, [IDRi], [IDRr],
                                         [IBAKE], V
```

4.2.2.1. Components of the I_MESSAGE_1 Message

The I_MESSAGE_1 message MUST always include the Header (HDR), Timestamp (T), and RAND payloads. The CSB ID (Crypto Session Bundle ID) SHALL be randomly selected by the Initiator. As the R_MESSAGE_1 message is mandatory, the Initiator indicates with the V flag that a verification message is expected.

The IDRi and IDRr payloads SHALL be included.

The IBAKE payload contains Initiator's Identity and EC Diffie-Hellman values (ECCPTi), and Responder's Identity all encrypted using Responder's Public Key (i.e., encr_key = K_PUBr) as follows:

IBAKE = E(encr_key, IDRi || ECCPTi || IDRr)

Optionally, Encrypted Secret Key (ESK) payload MAY be included. If included, ESK contains an identity and a Secret Key (SK) encrypted using intended Responder's Public Key (i.e., encr_key = K_PUBr).

ESK = E(encr_key, ID || SK)

4.2.2.2. Processing of the I_MESSAGE_1 Message

The parsing of I_MESSAGE_1 message SHALL be done as in [RFC3830]. If the received message is correctly parsed, the Responder SHALL use the Private Key (K_PRr) corresponding to the received IDRr to decrypt the IBAKE payload. If the message contains ESK payload, the Responder SHALL decrypt the SK and use it to decrypt the received IBAKE payload. Otherwise, if the Responder is not able to decrypt the

IBAKE payload, the Responder SHALL indicate it to the Initiator by including only its own EC Diffie-Hellman value (ECCPTr) in the next message (i.e., R_MESSAGE_1) it sends to the Initiator.

If the received message cannot be correctly parsed, the Responder SHOULD silently discard the message and abort the protocol.

4.2.2.3. Components of the R_MESSAGE_1 Message

The version, PRF func, CSB ID, #CS, and CS ID map type fields in the HDR payload SHALL be identical to the corresponding fields in the I_MESSAGE_1 message. The V flag SHALL be set to 1 as I_MESSAGE_2 message is mandatory.

The Timestamp type and value SHALL be identical to the one used in the I_MESSAGE_1 message.

The IDRi and IDRr payloads SHALL be included. The IDRi payload SHALL be as received in the I_MESSAGE_1. In the IDRr payload, the Responder SHALL include its own identity. Note that this identity might be different from the identity contained in the IDRr payload received in I_MESSAGE_1 message. The IDRr payloads of I_MESSAGE_1 and R_MESSAGE_1 will be different in the case of forking, retargeting, and deferred delivery.

The Responder's IBAKE payload contains the Initiator's EC Diffie-Hellman value (ECCPTi) received in I_MESSAGE_1 (if successfully decrypted), and the Initiator's EC Diffie-Hellman value generated by the Responder (ECCPTr), as well as corresponding Initiator and Responder's identities. If the Responder is unable to decrypt the IBAKE payload received in I_MESSAGE_1 (e.g., the message is received by the intended Responder's mailbox), the Responder SHALL include only its own EC Diffie-Hellman value (ECCPTr). The IBAKE payload in R_MESSAGE_1 is encrypted using Initiator's Public Key (i.e., encr_key = P_PUBi) as follows:

IBAKE = E(encr_key, IDRi || {ECCPTi} || IDRr || ECCPTr)

4.2.2.4. Processing of the R_MESSAGE_1 Message

The parsing of R_MESSAGE_1 message SHALL be done as in [RFC3830]. If the received message is correctly parsed, the Initiator shall use the Private Key corresponding to the received IDRi to decrypt the IBAKE payload. If the ECCPTi sent in I_MESSAGE_1 is not present in the received IBAKE payload (e.g., the Responder is currently offline and the R_MESSAGE_1 is received from Responder's mailbox), the Initiator

SHALL include ECCPTi again in the next message, I_MESSAGE_2. In this case, I_MESSAGE_2 SHALL also contain an ESK payload encrypted using the intended recipient's K_PUB.

If the received message cannot be correctly parsed, the Initiator SHOULD silently discard the message and abort the protocol.

4.2.2.5. Components of the I_MESSAGE_2 Message

The I_MESSAGE_2 message MUST always include the Header (HDR), Timestamp (T), and RAND payloads. The version, PRF func, CSB ID, #CS, and CS ID map type fields in the HDR payload SHALL be identical to the corresponding fields in the R_MESSAGE_2 message. As the R_MESSAGE_2 message is mandatory, the Initiator indicates with the V flag that a verification message is expected.

The IDRi and IDRr payloads SHALL be included. The IDRr payload SHALL be the same as the IDRr payload received in the R_MESSAGE_1.

The Initiator's IBAKE payload SHALL contain the Initiator's EC Diffie-Hellman value (ECCPTi) if the ECCPTi was not received in R_MESSAGE_1. Otherwise, ECCPTi SHALL NOT be included. The IBAKE payload in I_MESSAGE_2 SHALL contain the Initiator's and Responder's identities as well as Responder's EC Diffie-Hellman value received in message R_MESSAGE_1. IBAKE payload SHALL be encrypted using Responder's Public Key (i.e., encr_key = K_PUBr) as follows:

IBAKE = E(encr_key, IDRi || {ECCPTi} || IDRr || ECCPTr)

Optionally, Encrypted Secret Key (ESK) payload can be included. ESK SHALL be included in case R_MESSAGE_1 did not contain Initiator's EC Diffie-Hellman value (ECCPTi) (e.g., in the case of deferred delivery). If included, it contains an Initiator's identity and Initiator-generated Secret Key (SK) encrypted using intended recipient Public Key (i.e., encr_key = K_PUB) as follows:

ESK = E(encr_key, ID || SK)

4.2.2.6. Processing of the I_MESSAGE_2 Message

The parsing of the I_MESSAGE_2 message SHALL be done as in [RFC3830]. If the received message is correctly parsed, the Responder shall use the K_PRr corresponding to the received IDRr to decrypt the IBAKE payload. If an ESK is received, the Responder SHALL store it for future use (e.g., the Responder is a mailbox and will forward the key to the user once the user is online).

If the received message cannot be correctly parsed, the Responder SHOULD silently discard the message and abort the protocol.

4.2.2.7. Components of the R_MESSAGE_2 Message

The version, PRF func, CSB ID, #CS, and CS ID map type fields in the HDR payload SHALL be identical to the corresponding fields in the I_MESSAGE_2 message. The V flag SHALL be set to 0 by the Responder and ignored by the Initiator.

The Timestamp type and value SHALL be identical to the one used in the I_MESSAGE_2 message.

The IDRi and IDRr payloads SHOULD be included.

If Initiator's EC Diffie-Hellman value (ECCPTi) was received in I_MESSAGE_2, the Responder SHALL also include the IBAKE payload. If included, the IBAKE payload SHALL contain Initiator's EC Diffie-Hellman value (ECCPTi), and the Initiator's identity previously received in I_MESSAGE_2, encrypted using Initiator's Public Key (i.e., encr_key = K_PUBi) as follows:

IBAKE = E(encr_key, IDRi || ECCPTi)

The last payload SHALL be a Verification (V) payload where the authentication key (auth_key) is derived as specified in Section 5.2.

4.2.2.8. Processing of the R_MESSAGE_2 Message

The parsing of R_MESSAGE_2 message SHALL be done as in [RFC3830]. If the received message is correctly parsed, and if it contains the IBAKE payload, the Initiator SHALL use the K_PRi corresponding to the received IDRi to decrypt the IBAKE payload.

If the received message cannot be correctly parsed, the Initiator SHOULD silently discard the message and abort the protocol.

5. Key Management

The keys used in REQUEST_KEY_INIT/REQUEST_KEY_RESP exchange are derived from the pre-shared key or the envelope key as specified in [RFC3830]. As crypto sessions are not handled in this exchange, further keying material (i.e., TEKs) for this message exchange SHALL NOT be derived.

5.1. Generating Keys from the Session Key

As stated above, the session key [x][y]P is generated using exchanged EC Diffie-Hellman values, where x and y are randomly chosen by the Initiator and Responder. The session key, as a point on an elliptic curve, is then converted into octet string as specified in [SEC1]. This octet string K_SESSION is then used to generate MPK and TGK. Finally, the traffic encryption keys (e.g., TEK) are generated from TGK as specified in [RFC3830].

The MPK and TGK are generated from K_SESSION as follows.

```
inkey      : K_SESSION
inkey_len  : bit length of the MPK
label      : constant || 0xFF || 0xFFFFFFFF || RAND
outkey_len : desired bit length of the output key (MPK or TGK)
```

The constant depends on the derived key type as summarized below.

Derived Key	Constant
MPK	0x220E99A2
TGK	0x1F4D675B

Table 1: Constants for Key Derivation

The constants are taken from the decimal digits of e as described in [RFC3830].

5.2. Generating Keys for MIKEY Messages

The keys for MIKEY messages are used to protect the MIKEY messages exchanged between the Initiator and Responder (i.e., I_MESSAGE and R_MESSAGE). In the REQUEST_KEY_INIT/REQUEST_KEY_RESP exchange, the key derivation SHALL be done exactly as in [RFC3830].

MIKEY Protection Key (MPK) for I_MESSAGE/R_MESSAGE exchange is generated as described in Section 5.1. This MPK is then used to derive keys to protect R_MESSAGE_2 message.

```
inkey      : MPK
inkey_len  : bit length of the MPK
label      : constant || 0xFF || csb_id || RAND
outkey_len : desired bit length of the output key
```

where the constants are as defined in [RFC3830].

5.3. CSB Update

Similar to [RFC3830], MIKEY-IBAKE provides means for updating the CSB (Crypto Session Bundle), e.g., transporting new EC Diffe-Hellman values or adding new crypto sessions. The CSB updating is done by executing the exchange of I_MESSAGE_1/R_MESSAGE_1. The CSB updating MAY be started by either the Initiator or the Responder.

```
   Initiator                                  Responder

   I_MESSAGE_1 =                  ---->
   HDR, T, [IDRi], [IDRr],
      [IBAKE]                     <----       R_MESSAGE_1 =
                                              HDR, T, [IDRi], [IDRr],
                                              [IBAKE], V


   Responder                                  Initiator

   I_MESSAGE_1 =                  ---->
   HDR, T, [IDRr], [IDRi],
      [IBAKE]                     <----       R_MESSAGE_1 =
                                              HDR, T, [IDRi], [IDRr],
                                              [IBAKE], V
```

The new message exchange MUST use the same CSB ID as the initial exchange, but MUST use a new Timestamp. Other payloads that were provided in the initial exchange SHOULD NOT be included. New RANDs MUST NOT be included in the message exchange (the RANDs will only have effect in the initial exchange).

IBAKE payload with new EC Diffie-Hellman values SHOULD be included. If new EC Diffie-Hellman values are being exchanged during CSB updating, messages SHALL be protected with keys derived from EC Diffie-Hellman values exchanged as specified in Section 5.2. Otherwise, if new EC Diffie-Hellman values are not being exchanged during CSB update exchange, messages SHALL be protected with the keys that protected the I_MESSAGE/R_MESSAGE messages in the initial exchange.

5.4. Generating MAC and Verification Message

The authentication tag in all MIKEY-IBAKE messages is generated as described in [RFC3830]. As described above, the MPK is used to derive the auth_key. The MAC/Signature in the V/SIGN payloads covers the entire MIKEY message, except the MAC/Signature field itself and if there is an ESK payload in the massage it SHALL be omitted from MAC/Signature calculation. The identities (not whole payloads) of

the involved parties MUST directly follow the MIKEY message in the Verification MAC/Signature calculation. Note that in the I_MESSAGE/R_MESSAGE exchange, IDRr in R_MESSAGE_1 MAY not be the same as that appearing in I_MESSAGE_1.

6. Payload Encoding

This section does not describe all the payloads that are used in the new message types. It describes in detail the new IBAKE and ESK payloads and in less detail the payloads for which changes has been made compared to [RFC3830]. For a detailed description of the MIKEY payloads (e.g., Timestamp (T) payload, RAND payload, etc.), see [RFC3830]. For the description of IDR payload as well as for the definition of additional PRF functions and encryption algorithms not defined in [RFC3830], see [RFC6043].

6.1. Common Header Payload (HDR)

For the Common Header Payload, new values are added to the data type and the next payload namespaces.

- Data type (8 bits): describes the type of message.

Data Type	Value	Comment
REQUEST_KEY_PSK	19	Request Private Keys message (PSK)
REQUEST_KEY_PKE	20	Request Private Keys message (PKE)
REQUEST_KEY_RESP	21	Response Private Keys message
I_MESSAGE_1	22	First Initiator's message
R_MESSAGE_1	23	First Responder's message
I_MESSAGE_2	24	Second Initiator's message
R_MESSAGE_2	25	Second Responder's message

Table 2: Data Type (Additions)

- Next payload (8 bits): identifies the payload that is added after this payload.

Next Payload	Value	Section
IBAKE	22	Section 6.1.1
ESK	23	Section 6.1.2
SK	24	Section 6.1.5
ECCPT	25	Section 6.1.4

Table 3: Next Payload (Additions)

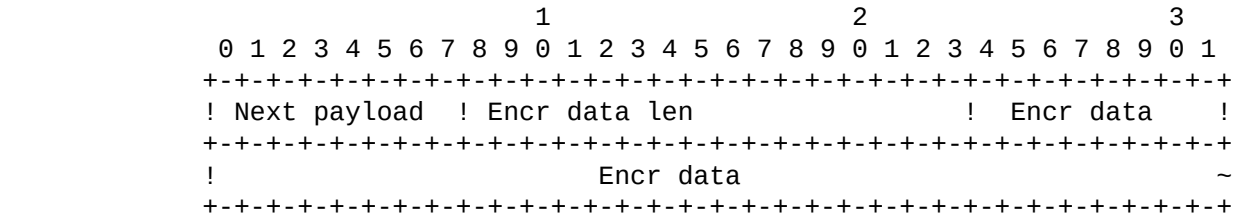
- V (1 bits): flag to indicate whether or not a response message is expected (this only has meaning when it is set in an initiation message). If a response is required, the V flag SHALL always be set to 1 in the initiation messages and the receiver of the initiation message (Responder or KMS) SHALL ignore it.

- #CS (8 bits): indicates the number of crypto sessions that will be handled within the CSB. It SHALL be set to 0 in the Request Key exchange, as crypto sessions SHALL NOT be handled.

- CS ID map type (8 bits): specifies the method of uniquely mapping crypto sessions to the security protocol sessions. In the Request Key exchange, the CS ID map type SHALL be the "Empty map" (defined in [RFC4563]) as crypto sessions SHALL NOT be handled.

6.1.1. IBAKE Payload

The IBAKE payload contains IBE encrypted (see [RFC5091] and [RFC5408] for details about IBE) Initiator and Responder's Identities and EC Diffie-Hellman Sub-Payloads (see Section 6.1.4 for the definition of EC Diffie-Hellman Sub-Payload). It may contain one or more EC Diffie-Hellman Sub-Payloads and their associated identities. The last EC Diffie-Hellman or Identity Sub-Payload has its Next payload field set to Last payload.



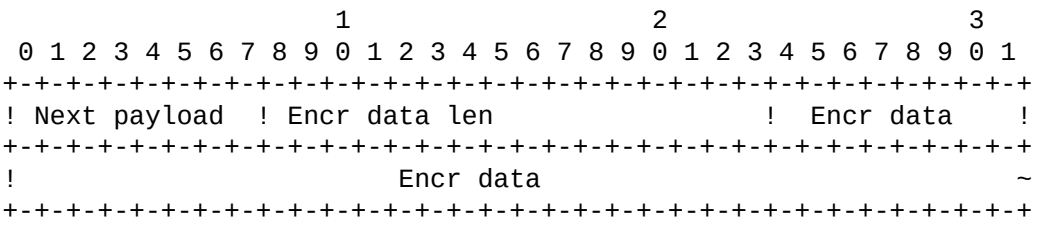
- Next payload (8 bits): identifies the payload that is added after this payload.

- Encr data len (16 bits): length of Encr data (in bytes).

- Encr data (variable length): the IBE encrypted EC Diffie-Hellman Sub-Payloads (see Section 6.1.4) and their associated Identity payloads.

6.1.2. Encrypted Secret Key (ESK) Payload

The Encrypted Secret Key payload contains IBE encrypted (see [RFC5091] and [RFC5408] for details about IBE) Secret Key Sub-Payload and its associated identity (see Section 6.1.5 for the definition of the Secret Key Sub-Payload).



- Next payload (8 bits): identifies the payload that is added after this payload.

- Encr data len (16 bits): length of Encr data (in bytes).

- Encr data (variable length): the encrypted secret key Sub-Payloads (see Section 6.1.5).

6.1.3. Key Data Sub-Payload

For the key data Sub-Payload, a new type of key is defined. The Private Key (K_PR) is used to decrypt the content encrypted using the corresponding Public Key (K_PUB). KEMAC in the REQUEST_KEY_RESP SHALL contain one or more Private Keys.

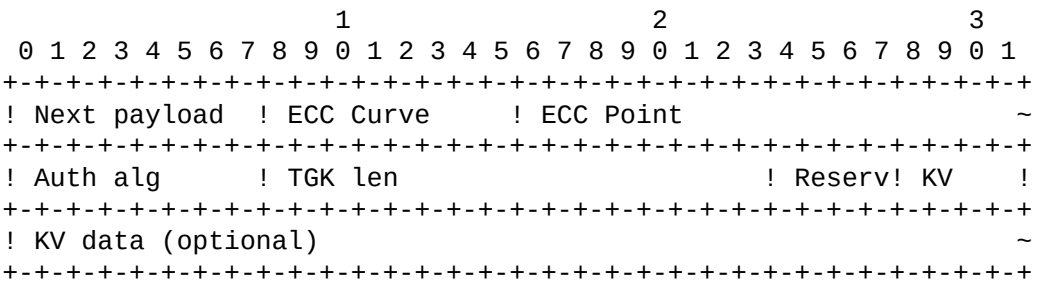
- Type (4 bits): indicates the type of key included in the payload.

Type	Value	Comments
K_PR	7	Private Key

Table 4: Key Data Type (Additions)

6.1.4. EC Diffie-Hellman Sub-Payload

The EC Diffie-Hellman (ECCPT) Sub-Payload uses the format defined below. The EC Diffie-Hellman Sub-Payload in MIKEY-IBAKE is never included in clear, but as an encrypted part of the IBAKE payload.



- Next payload (8 bits): identifies the payload that is added after this payload. See Section 6.1 of [RFC3830] for values.

- ECC curve (8 bits): identifies the ECC curve used.

ECC Curve	Value
ECPRGF192Random / P-192 / secp192r1	1
EC2NGF163Random / B-163 / sect163r2	2
EC2NGF163Koblitz / K-163 / sect163k1	3
EC2NGF163Random2 / none / sect163r1	4
ECPRGF224Random / P-224 / secp224r1	5
EC2NGF233Random / B-233 / sect233r1	6
EC2NGF233Koblitz / K-233 / sect233k1	7
ECPRGF256Random / P-256 / secp256r1	8
EC2NGF283Random / B-283 / sect283r1	9
EC2NGF283Koblitz / K-283 / sect283k1	10
ECPRGF384Random / P-384 / secp384r1	11
EC2NGF409Random / B-409 / sect409r1	12
EC2NGF409Koblitz / K-409 / sect409k1	13
ECPRGF521Random / P-521 / secp521r1	14
EC2NGF571Random / B-571 / sect571r1	15
EC2NGF571Koblitz / K-571 / sect571k1	16

Table 5: Elliptic Curves

- ECC point (variable length): ECC point data, padded to end on a 32-bit boundary, encoded in octet string representation.

- Auth alg (8 bits): specifies the MAC algorithm used for the verification message. For MIKEY-IBAKE this field is ignored.

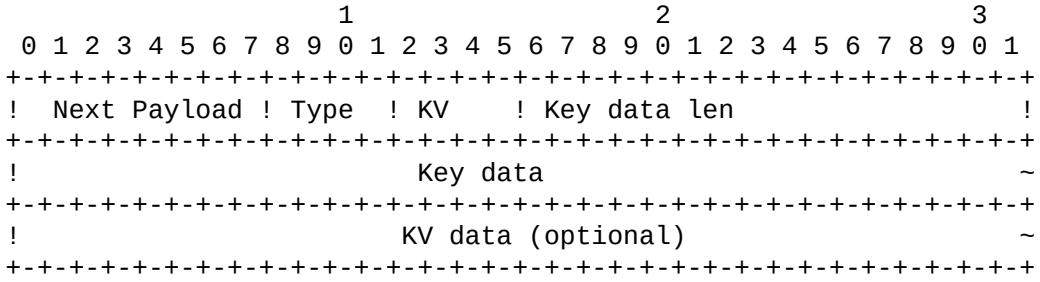
- TGK len (16 bits): the length of the TGK (in bytes). For MIKEY-IBAKE this field is ignored.

- KV (4 bits): indicates the type of key validity period specified. This may be done by using an SPI (alternatively an MKI in SRTP) or by providing an interval in which the key is valid (e.g., in the latter case, for SRTP this will be the index range where the key is valid). See Section 6.13 of [RFC3830] for pre-defined values.

- KV data (variable length): This includes either the SPI/MKI or an interval (see Section 6.14 of [RFC3830]). If KV is NULL, this field is not included.

6.1.5. Secret Key Sub-Payload

Secret Key payload is included as a Sub-Payload in Encrypted Secret Key payload. Similar to EC Diffie-Hellman Sub-Payload, it is never included in clear, but as an encrypted part of the ESK payload.



- Next payload (8 bits): identifies the payload that is added after this payload.

- Type (4 bits): indicates the type of the key included in the payload.

Type	Value
SK	1

Table 6: Secret Key Types

- KV (4 bits): indicates the type of key validity period specified. This may be done by using an SPI (or MKI in the case of [RFC3711]) or by providing an interval in which the key is valid (e.g., in the latter case, for SRTP this will be the index range where the key is valid). KV values are the same as in Section 6.13 of [RFC3830]

- Key data len (16 bits): the length of the Key data field (in bytes).

- Key data (variable length): The SK data.

- KV data (variable length): This includes either the SPI or an interval. If KV is NULL, this field is not included.

7. Security Considerations

Unless explicitly stated, the security properties of the MIKEY protocol as described in [RFC3830] apply to MIKEY-IBAKE as well. In addition, MIKEY-IBAKE is based on the basic Identity-Based Encryption protocol, as specified in [RFC5091], [RFC5408], and [RFC5409], and as such inherits some properties of that protocol. For instance, by concatenating the "date" with the identity (to derive the Public Key), the need for any key revocation mechanisms is virtually eliminated. Moreover, by allowing the participants to acquire multiple Private Keys (e.g., for duration of contract) the availability requirements on the KMS are also reduced without any reduction in security.

7.1. General Security Considerations

The MIKEY-IBAKE protocol relies on the use of Identity-Based Encryption. [RFC5091] describes attacks on the cryptographic algorithms used in Identity-Based Encryption. In addition, [RFC5091] provides recommendations for security parameters for described IBE algorithms.

It is assumed that the Key Management Services are secure, not compromised, trusted, and will not engage in launching active attacks independently or in a collaborative environment. However, any malicious insider could potentially launch passive attacks (by decryption of one or more message exchanges offline). While it is in the best interest of administrators to prevent such attacks, it is hard to eliminate this problem. Hence, it is assumed that such problems will persist, and hence the protocols are designed to protect participants from passive adversaries.

7.2. IBAKE Protocol Security Considerations

For the basic IBAKE protocol, from a cryptographic perspective, the following security considerations apply.

In every step, Identity-Based Encryption (IBE) is used with the recipient's Public Key. This guarantees that only the intended recipient of the message can decrypt the message [BF].

Next, the use of identities within the encrypted payload is intended to eliminate some basic reflection attacks. For instance, suppose identities were not used as part of the encrypted payload, in the first step of the IBAKE protocol (i.e., I_MESSAGE_1 of Figure 3 in Section 4.1). Furthermore, assume an adversary who has access to the conversation between Initiator and Responder and can actively snoop into packets and drop/modify them before routing them to the destination. For instance, assume that the IP source address and destination address can be modified by the adversary. After the first message is sent by the Initiator (to the Responder), the adversary can take over and trap the packet. Next, the adversary can modify the IP source address to include adversary's IP address, before routing it onto the Responder. The Responder will assume the request for an IBAKE session came from the adversary and will execute step 2 of the IBAKE protocol (i.e., R_MESSAGE_1 of Figure 3 in Section 4.1) but encrypt it using the adversary's Public Key. The above message can be decrypted by the adversary (and only by the adversary). In particular, since the second message includes the challenge sent by the Initiator to the Responder, the adversary will now learn the challenge sent by the Initiator. Following this, the adversary can carry on a conversation with the Initiator "pretending" to be the Responder. This attack will be eliminated if identities are used as part of the encrypted payload. In summary, at the end of the exchange both Initiator and Responder can mutually authenticate each other and agree on a session key.

Recall that Identity-Based Encryption guarantees that only the recipient of the message can decrypt the message using the Private Key. The caveat being, the KMS that generated the Private Key of recipient of message can decrypt the message as well. However, the KMS cannot learn the session key [x][y]P given [x]P and [y]P based on the Elliptic Curve Diffie-Hellman problem. This property of resistance to passive key escrow from the KMS is not applicable to the basic IBE protocols proposed in [RFC5091], [RFC5408], and [RFC5409].

Observe that the protocol works even if the Initiator and Responder belong to two different Key Management Services. In particular, the parameters used for encryption to the Responder and parameters used

for encryption to the Initiator can be completely different and independent of each other. Moreover, the Elliptic Curve used to generate the session key [x][y]P can be completely different. If such flexibility is desired, then it would be advantageous to add optional extra data to the protocol to exchange the algebraic primitives used in deriving the session key.

In addition to mutual authentication, and resistance to passive escrow, the Diffie-Hellman property of the session key exchange guarantees perfect secrecy of keys. In others, accidental leakage of one session key does not compromise past or future session keys between the same Initiator and Responder.

7.3. Forking

In the Forking feature, given that there are multiple potential Responders, it is important to observe that there is one "common Responder" identity (and corresponding Public and Private Keys) and each Responder has a unique identity (and corresponding Public and Private Keys). Observe that, in this framework, if one Responder responds to the invite from the Initiator, it uses its unique identity such that the protocol guarantees that no other Responder learns the session key.

7.4. Retargeting

In the Retargeting feature, the forwarding server does not learn the Private Key of the intended Responder since it is encrypted using the retargeted Responder's Public Key. Additionally, the Initiator will learn that the retargeted Responder answered the phone (and not the intended Responder) since the retargeted Responder includes its own identity in the message sent to the Initiator. This will allow the Initiator to decide whether or not to carry on the conversation. Finally, the session key cannot be discovered by the intended Responder since the random number chosen by the retargeted Responder is not known to the intended Responder.

7.5. Deferred Delivery

In the Deferred Delivery feature, the Initiator and the Responder's mailbox will mutually authenticate each other thereby preventing server side "phishing" attacks and conversely guarantees to the server (and eventually to the Responder) the identity of the Initiator. Moreover, the key used by Initiator to encrypt the contents of the message is completely independent from the session key derived between the Initiator and the server. Finally, the key

used to encrypt the message is encrypted using the Responder's Public Key, which allows the contents of the message to remain unknown to the mailbox server.

8. IANA Considerations

This document defines several new values for the namespaces Data Type, Next Payload, and Key Data Type defined in [RFC3830]. The following IANA assignments have been added to the MIKEY Payload registry (in bracket is a reference to the table containing the registered values):

- Data Type (see Table 2)
- Next Payload (see Table 3)
- Key Data Type (see Table 4)

The ECCPT payload defines an 8-bit ECC Curve field for which IANA has created and will maintain a new namespace in the MIKEY Payload registry. Assignments consist of an ECC curve and its associated value. Values in the range 1-239 SHOULD be approved by the process of Specification Required, values in the range 240-254 are for Private Use, and the values 0 and 255 are Reserved according to [RFC5226]. The initial contents of the registry are as follows:

Value	ECC curve
0	Reserved
1	ECPRGF192Random / P-192 / secp192r1
2	EC2NGF163Random / B-163 / sect163r2
3	EC2NGF163Koblitz / K-163 / sect163k1
4	EC2NGF163Random2 / none / sect163r1
5	ECPRGF224Random / P-224 / secp224r1
6	EC2NGF233Random / B-233 / sect233r1
7	EC2NGF233Koblitz / K-233 / sect233k1
8	ECPRGF256Random / P-256 / secp256r1
9	EC2NGF283Random / B-283 / sect283r1
10	EC2NGF283Koblitz / K-283 / sect283k1
11	ECPRGF384Random / P-384 / secp384r1
12	EC2NGF409Random / B-409 / sect409r1
13	EC2NGF409Koblitz / K-409 / sect409k1
14	ECPRGF521Random / P-521 / secp521r1
15	EC2NGF571Random / B-571 / sect571r1
16	EC2NGF571Koblitz / K-571 / sect571k1
17-239	Unassigned
240-254	Private Use
255	Reserved

The SK Sub-Payload defines a 4-bit Type field for which IANA has created and will maintain a new namespace in the MIKEY Payload registry. Assignments consist of a type of key and its associated value. Values in the range 2-15 SHOULD be approved by the process of Specification Required. The initial contents of the registry are as follows:

Value	Type
0	Reserved
1	Secret Key (SK)
2-15	Unassigned

9. References

9.1. Normative References

[BF] Boneh, D. and M. Franklin, "Identity-Based Encryption from the Weil Pairing", in SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

[RFC4563] Carrara, E., Lehtovirta, V., and K. Norrman, "The Key ID Information Type for the General Extension Payload in Multimedia Internet KEYing (MIKEY)", RFC 4563, June 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC6043] Mattsson, J. and T. Tian, "MIKEY-TICKET: Ticket-Based Modes of Key Distribution in Multimedia Internet KEYing (MIKEY)", RFC 6043, March 2011.

[SEC1] Standards for Efficient Cryptography Group, "Elliptic Curve Cryptography", September 2000.

9.2. Informative References

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

[RFC4650] Euchner, M., "HMAC-Authenticated Diffie-Hellman for Multimedia Internet KEYing (MIKEY)", RFC 4650, September 2006.

[RFC4738] Ignjatic, D., Dondeti, L., Audet, F., and P. Lin, "MIKEY-RSA-R: An Additional Mode of Key Distribution in Multimedia Internet KEYing (MIKEY)", RFC 4738, November 2006.

[RFC5091] Boyen, X. and L. Martin, "Identity-Based Cryptography Standard (IBCS) #1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems", RFC 5091, December 2007.

[RFC5408] Appenzeller, G., Martin, L., and M. Schertler, "Identity-Based Encryption Architecture and Supporting Data Structures", RFC 5408, January 2009.

[RFC5409] Martin, L. and M. Schertler, "Using the Boneh-Franklin and Boneh-Boyen Identity-Based Encryption Algorithms with the Cryptographic Message Syntax (CMS)", RFC 5409, January 2009.

Authors' Addresses

Violeta Cakulev
Alcatel Lucent
600 Mountain Ave.
3D-517
Murray Hill, NJ 07974
US

Phone: +1 908 582 3207
EMail: violeta.cakulev@alcatel-lucent.com

Ganapathy Sundaram
Alcatel Lucent
600 Mountain Ave.
3D-517
Murray Hill, NJ 07974
US

Phone: +1 908 582 3209
EMail: ganesh.sundaram@alcatel-lucent.com