Groupe de travail Réseau
Request for Comments : 3830
Catégorie : En cours de normalisation
août 2004
Traduction Claude Brière de l'Isle
Mise à jour par la RFC4738

J. Arkko
E. Carrara
F. Lindholm
M. Naslund
K. Norrman, Ericsson Research

MIKEY : Gestion de clé multimédia pour l'Internet

Résumé
Le présent document décrit un schéma de gestion de clés qui peut être utilisé pour les applications en temps réel (aussi bien pour les communications d'homologue à homologue que pour les communications de groupe). En particulier, son utilisation pour prendre en charge le protocole sécurisé de transport en temps réel est décrite en détail.

Les protocoles de sécurité pour les applications multimédia en temps réel ont commencé à apparaître. Cela a mis en avant le besoin d'une solution de gestion de clés pour soutenir ces protocoles.

Table des Matières

1. Introduction

Des travaux récents ont défini un protocole de sécurité pour la protection des applications en temps réel fonctionnant sur RTP, [RFC3711]. Cependant, un protocole de sécurité a besoin d'une solution de gestion de clés pour échanger les clés et les paramètres de sécurité qui s'y rapportent. Il y a des propriétés fondamentales qu'un tel schéma de gestion de clés doit satisfaire pour servir les applications en direct et en temps réel (telles que l'envoi individuel et la diffusion groupée) en particulier dans les réseaux hétérogènes (mélange de filaire et de sans fil).

Le présent document décrit une solution de gestion de clés qui s'adresse aux scénarios multimédia (par exemple, les appels SIP [RFC3261] et les sessions RTSP [RFC2326]). L'accent est mis sur la façon d'établir la gestion de clés pour des sessions multimédia sûres telles que soient satisfaites les exigences dans un environnement hétérogène.

1.1 Solutions existantes

L'IETF travaille à développer des schémas de gestion de clés. Par exemple, IKE [RFC2409] est un schéma largement accepté d'envoi individuel pour IPsec, et le groupe de travail MSEC développe d'autres schémas pour traiter la communication de groupe [RFC3447], [RFC4535]. Cependant, pour des raisons qu'on exposera plus loin, il y a besoin d'un schéma avec une plus faible latence, convenant à des cas exigeants comme celui des données en temps réel sur des réseaux hétérogènes et des petits groupes interactifs.

Dans certains cas, une option pourrait être d'utiliser la [RFC2327], car SDP définit un champ pour le transport des clés, le champ "k=". Cependant, ce champ ne peut pas être utilisé pour les besoins plus généraux de la gestion de clés, car il ne peut pas être étendu à partir de la définition actuelle.

1.2 Conventions de notation

Les mots clés "DOIT", "NE DOIT PAS", "EXIGÉ", "DEVRA", "NE DEVRA PAS", "DEVRAIT", "NE DEVRAIT PAS", "RECOMMANDÉ", "PEUT", et "FACULTATIF" dans ce document sont à interpréter comme décrit dans le BCP 14, [RFC2119].

1.3 Définitions

Protocole de sécurité (de données) : c'est le protocole de sécurité utilisé pour protéger le trafic réel de données. Des exemples de protocoles de sécurité sont IPsec et SRTP.

Association de sécurité des données (SA, *Security Association*) : informations pour le protocole de sécurité, incluant une TEK et un ensemble de paramètres/politiques.

Session de chiffrement (CS, *Crypto Session*) : flux de données uni ou bidirectionnel, protégé par une seule instance de protocole de sécurité. Par exemple, lorsque SRTP est utilisé, la session de chiffrement va souvent contenir deux flux, un flux RTP et le flux RTCP correspondant, qui sont tous deux protégés par un seul contexte cryptographique SRTP, c'est-à-dire, qu'ils partagent les données de clé et le gros des paramètres de sécurité dans le contexte cryptographique SRTP (comportement par défaut dans la [RFC3711]). Dans le cas d'IPsec, une session de chiffrement représenterait une instance de SA IPsec. Une session de chiffrement peut être vue comme une SA de données (comme défini dans la [RFC4046]) et pourrait donc être transposée si nécessaire à d'autres protocoles de sécurité.

Faisceau de session de chiffrement (CSB, *Crypto Session Bundle*) : collection d'une ou plusieurs sessions de chiffrement, qui peuvent avoir des TGK (voir ci-dessous) et paramètres de sécurité communs.

Identifiant de session de chiffrement : identifiant univoque pour la CS au sein d'une CSB.

Identifiant de faisceau de session de chiffrement (CSB ID) : identifiant univoque de ce CSB.

Clé de génération de TEK (TGK, *TEK Generation Key*) : chaîne binaire sur laquelle deux parties se sont mises d'accord, associée au CSB. À partir de la TGK, les clés de chiffrement de trafic peuvent alors être générées sans avoir besoin d'autre communication.

Clé de chiffrement de trafic (TEK, *Traffic-Encrypting Key*) : c'est la clé utilisée par le protocole de sécurité pour protéger la CS (cette clé peut être utilisée directement par le protocole de sécurité ou peut être utilisée pour déduire d'autres clés selon le protocole de sécurité). Les TEK sont déduites de la TGK du CSB.

Changement de clé de TGK : c'est le processus de renégociation/mise à jour de la TGK (et par conséquent, des futures TEK).

Initiateur : c'est l'initiateur du protocole de gestion de clé, pas nécessairement l'initiateur de la communication.

Répondeur : c'est celui qui répond dans le protocole de gestion de clés.

Clé de salage : chaîne aléatoire ou pseudo-aléatoire (voir [RFC1750], [HAC]) utilisée pour protéger contre certaines attaques de pré-calcul hors ligne sur le protocole de sécurité sous-jacent.

PRF(k,x) : fonction de chiffrement pseudo-aléatoire (voir [HAC]).
E(k,m) : chiffrement de m avec la clé k.
PKx : clé publique de x
[] : élément d'information facultatif
{} : note zéro, une ou plusieurs occurrences
|| : enchaînement
| : OU (opérateur de tri)
^ : exponentiation
OUX : ou exclusif

Ordre des bits et des octets : tout au long de ce document, les bits et les octets sont indexés, comme d'habitude, de gauche à droite, les bits les plus à gauche étant ceux de plus fort poids.

1.4. Abréviations

AES *(Advanced Encryption Standard)* norme de chiffrement évolué
CM *(Counter Mode)* mode compteur (comme défini dans la [RFC3711])
CS *(Crypto Session)* session de chiffrement
CSB *(Crypto Session Bundle)* faisceau de session de chiffrement
DH Diffie-Hellman
DoS *(Denial of Service)* déni de service
MAC *(Message Authentication Code)* code d'authentification de message
MIKEY *(Multimedia Internet KEYing)* gestion de clé multimédia pour l'Internet
PK *(Public-Key)* clé publique
PKE *(Envelope Key Payload)* charge utile de clé enveloppe
PSK *(Pre-Shared key)* clé pré-partagée
RTP *(Real-time Transport Protocol)* protocole de transport en temps réel
RTSP *(Real Time Streaming Protocol)* protocole de direct en temps réel
SDP *(Session Description Protocol)* protocole de description de session
SIP *(Session Initiation Protocol)* protocole d'initiation de session
SRTP *(Secure RTP)* RTP sécurisé
TEK *(Traffic-encrypting key)* clé de chiffrement de trafic
TGK *(TEK Generation Key)* clé de génération de clé de chiffrement de trafic

1.5 Plan du document

La Section 2 décrit le scénario de base et les objectifs de conception auxquels MIKEY est destiné. Elle donne aussi une brève vue d'ensemble de la solution entière et de sa relation à l'architecture de gestion de clé de groupe [RFC4046].

Les mécanismes de base de transport/échange de clé sont expliqués en détail à la Section 3. La déduction de clé et les autres procédures générales de gestion de clé sont décrites à la Section 4.

La Section 5 décrit le comportement attendu des parties. Cela inclut aussi la création et l'analyse des messages.

Toutes les définitions des charges utiles de MIKEY sont décrites à la Section 6.

La Section 7 traite des considérations de transport, tandis que la Section 8 se concentre sur la façon dont MIKEY est utilisé dans les scénarios de groupe.

La section des considérations pour la sécurité (Section 9) donne une explication plus en profondeur des importantes questions qui se rapportent à la sécurité

2. Vue d'ensemble basique

2.1 Scénarios

MIKEY est principalement destiné à être utilisé pour de l'homologue à homologue, du simple d'un à plusieurs, et des groupes de petite taille (interactifs). Un des principaux scénarios multimédia considérés lors de la conception de MIKEY a été le scénario de conversation multimédia, où les usagers peuvent interagir et communiquer en temps réel. Dans ces scénarios, on peut s'attendre à ce que les homologues établissent des sessions multimédia les uns avec les autres, où une session multimédia peut consister en un ou plusieurs flux multimédia sécurisés (par exemple, des flux SRTP).

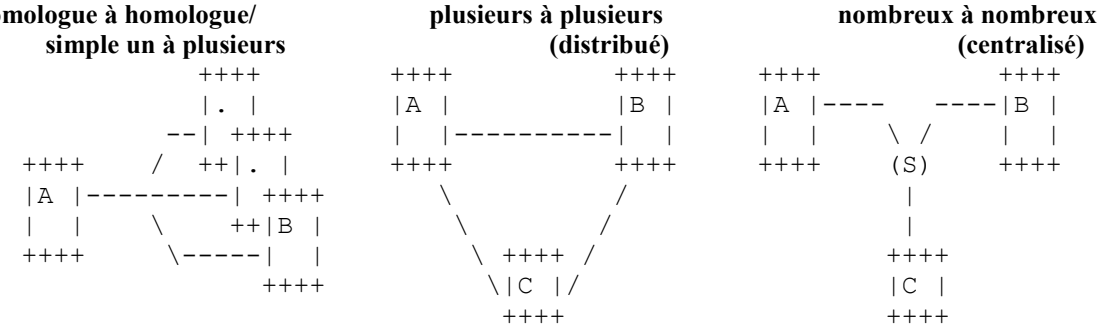


Figure 2.1 : Exemples des quatre scénarios : homologue à homologue, simple un à plusieurs, plusieurs à plusieurs sans serveur centralisé (aussi noté petit groupe interactif) **et nombreux à nombreux avec serveur centralisé**

On identifie dans ce qui suit des scénarios typiques qui impliquent les applications multimédia dont nous traitons (voir aussi la Figure 2.1).

a) d'homologue à homologue (en envoi individuel) par exemple, un appel fondé sur SIP [RFC3261] entre deux parties, où il peut être souhaitable que la sécurité soit établie par accord mutuel ou bien que chaque partie établisse la sécurité pour ses propres flux sortants.

b) simple d'un à plusieurs (en diffusion groupée) par exemple, des présentations en temps réel, où l'envoyeur est chargé d'établir la sécurité.

c) de plusieurs à plusieurs, sans unité de contrôle centralisée, par exemple, pour de petits groupes interactifs où chaque partie peut établir la sécurité pour ses propres supports de sortie. Deux modèles de base peuvent être utilisés ici. Dans le premier modèle, l'initiateur du groupe agit comme serveur du groupe (et est le seul autorisé à inclure de nouveaux membres). Dans le second modèle, les informations d'autorisation d'inclure de nouveaux membres peuvent être déléguées aux autres participants.

d) de nombreux à nombreux, avec une unité de contrôle centralisée, par exemple, pour de plus grands groupes avec une sorte de contrôleur de groupe qui établit la sécurité.

Les solutions de gestion de clés peuvent être différentes dans les scénarios ci-dessus. Lors de la conception de MIKEY, l'accent a été mis sur les cas a, b, et c. Pour le scénario c, seul le premier modèle est couvert par le présent document.

2.2 Objectifs de conception

Le protocole de gestion de clés est conçu comme ayant les caractéristiques suivantes :

* Sécurité de bout en bout. Seuls les participants impliqués dans la communication ont accès aux clés générées.
* Simplicité.
* Efficacité. Conçu pour avoir :
  - une faible consommation de bande passante,
  - une faible charge de calcul,
  - une petite taille de code, et
  - un nombre minimal d'allers-retours.
* Tunnelage. Possibilité de "tunneler"/intégrer MIKEY dans les protocoles d'établissement de session (par exemple, SDP et RTSP).
* Indépendance à toute fonctionnalité de sécurité spécifique du transport sous-jacent.

2.3 Vue d'ensemble du système

Un objectif de MIKEY est de produire une SA de données pour le protocole de sécurité, incluant une clé de chiffrement du trafic (TEK, *traffic-encrypting key*) qui est déduite d'une clé de génération de TEK (TGK, *TEK Generation Key*), et utilisée comme entrée pour le protocole de sécurité.

MIKEY prend en charge la possibilité d'établir des clés et paramètres pour plus d'un protocole de sécurité (ou pour plusieurs instances du même protocole de sécurité) en même temps. Le concept de faisceau de session de chiffrement (CSB, *Crypto Session Bundle*) est utilisé pour noter une collection d'une ou plusieurs sessions de chiffrement qui peuvent avoir en commun TGK et paramètres de sécurité, mais qui obtiennent des TEK distinctes de MIKEY.

La procédure d'établissement d'une CSB et de création d'une TEK (et SA de données) est faite selon la Figure 2.2 :

1. Un ensemble de paramètres de sécurité et de TGK est accepté pour le faisceau de sessions de chiffrement (cela se fait par l'un des trois mécanismes de transport/échange de clés proposés, voir la Section 3).

2. Les TGK sont utilisées pour déduire (d'une façon cryptographiquement sûre) une TEK pour chaque session de chiffrement.

3. La TEK, avec les paramètres du protocole de sécurité, représente la SA de données, qui est utilisée comme entrée du protocole de sécurité.

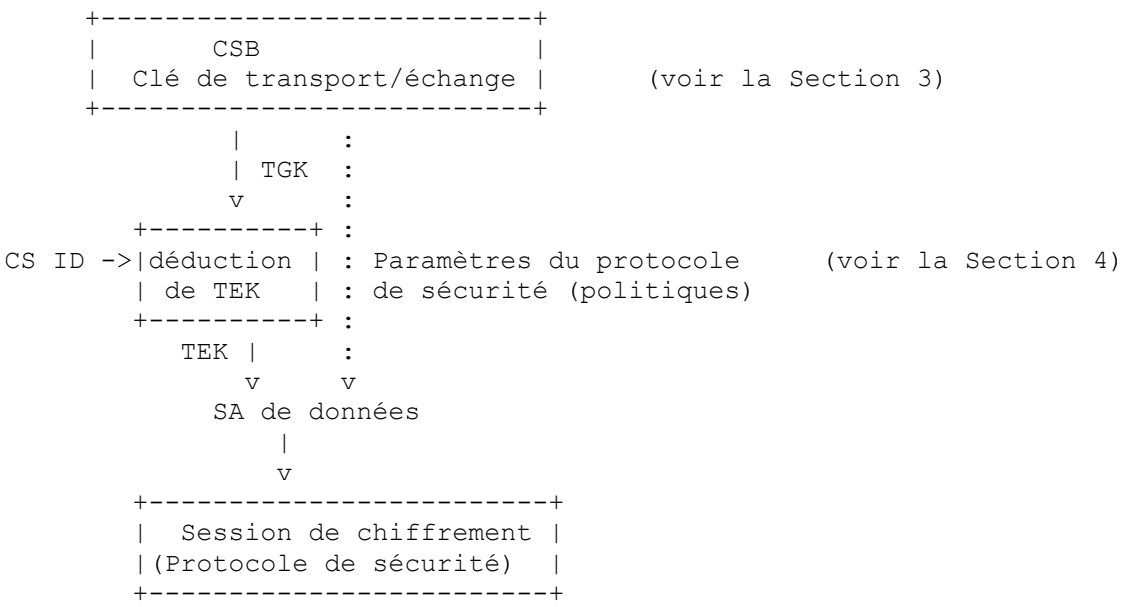


Figure 2.2 : Vue d'ensemble des procédures de gestion de clé de MIKEY

Le protocole de sécurité peut alors utiliser directement la TEK, ou, si il l'accepte, déduire d'autres clés de session de la TEK (par exemple, voir SRTP [RFC3711]). Il dépend cependant du protocole de sécurité de définir comment la TEK est utilisée.

MIKEY peut être utilisé pour mettre à jour les TEK et les sessions de chiffrement dans un faisceau de session de chiffrement courante (voir au paragraphe 4.5). Cela se fait en exécutant la phase transport/échange une fois de plus pour obtenir une nouvelle TGK (et par conséquent déduire de nouvelles TEK) ou pour mettre à jour d'autres paramètres spécifiques de la session de chiffrement.

2.4 Relations avec GKMARCH

L'architecture de gestion de clé de groupe (GKMARCH, *Group key management architecture*) [RFC4046] décrit une architecture générale pour les protocoles de gestion de clé de groupe. MIKEY fait partie de cette architecture, et peut être utilisé comme ce qu'on appelle un protocole d'enregistrement. Les principales entités impliquées dans cette architecture sont le contrôleur de groupe/serveur de clés (GCKS, *group controller/key server*) le ou les receveurs, et le ou les envoyeurs.

Dans MIKEY, l'envoyeur pourrait agir comme GCKS et pousser les clés vers le ou les receveurs.

Noter que, par exemple, dans un appel initié par SIP, l'envoyeur peut aussi être un receveur. Comme MIKEY s'adresse à de petits groupes interactifs, un membre peut passer de façon dynamique de l'état d'envoyeur à celui de receveur (ou être les deux simultanément).

3. Méthodes de base de transport et d'échange de clés

Les paragraphes qui suivent définissent trois différentes méthodes de transport/établissement d'une TGK : avec l'utilisation d'une clé prépartagée, le chiffrement par clé publique, et l'échange de clés Diffie-Hellman (DH). Dans ce qui suit, on suppose, pour simplifier, une communication en envoi individuel. En plus de la TGK, un "nom occasionnel" aléatoire, noté RAND, est aussi transporté. Dans les trois cas, les valeurs de TGK et de RAND sont alors utilisées pour déduire les TEK comme décrit au paragraphe 4.1.3. Un horodatage est aussi envoyé pour éviter des attaques en répétition (voir au paragraphe 5.4).

La méthode des clés prépartagées et la méthode des clés publiques sont toutes deux fondées sur des mécanismes de transport de clés, où la TGK réelle est poussée (de façon sûre) aux receveurs. Dans la méthode Diffie-Hellman, la TGK réelle est plutôt déduite de valeurs Diffie-Hellman échangées entre les homologues.

Le cas prépartagé est, de loin, la façon la plus efficace de traiter le transport de clés à cause de l'utilisation de cryptographie uniquement symétrique. Cette approche présente aussi l'avantage que seule une petite quantité de données doit être échangée. Bien sûr, la question problématique est l'adaptabilité car il n'est pas toujours faisable de partager des clés

individuelles avec un grand groupe d'homologues. Donc, ce cas s'adresse principalement aux scénarios tels que de serveur à client et aussi les cas où les modes de clé publique ont déjà été utilisés, permettant donc la "mise en antémémoire" d'une clé symétrique (voir ci-dessous et au paragraphe 3.2).

La cryptographie à clé publique peut être utilisée pour créer un système adaptable. Un inconvénient de cette approche est qu'elle consomme plus de ressources que celle de la clé prépartagée. Un autre inconvénient est que dans la plupart des cas, une infrastructure de clé publique (PKI, *Public Key Infrastructure*) est nécessaire pour traiter la distribution des clés publiques. Bien sûr, il est possible d'utiliser des clés publiques comme clés prépartagées (par exemple, en utilisant des certificats auto signés). On devrait aussi noter que, comme on l'a mentionné plus haut, cette méthode peut être utilisée pour établir une clé symétrique "en antémémoire" qui peut être utilisée ultérieurement pour établir des TGK en utilisant la méthode des clés prépartagées (et donc, la demande suivante peut être exécutée plus efficacement).

En général, la méthode de l'accord de clé Diffie-Hellman (DH) consomme plus de ressources (à la fois en calcul et en bande passante) que les précédentes, et a besoin de certificats comme dans le cas des clés publiques. Cependant, elle a l'avantage de fournir un secret de transmission parfait (PFS, *perfect forward secrecy*) et de la souplesse en permettant la mise en œuvre dans plusieurs groupes finis différents.

Noter qu'en utilisant la méthode DH, les deux parties impliquées vont générer une clé aléatoire unique imprévisible. Donc, il n'est pas possible d'utiliser cette méthode DH pour établir une TEK de groupe (car les différentes parties du groupe finiraient avec des TEK différentes). L'intention de la méthode DH n'est pas de travailler sur ce scénario, mais d'être une bonne alternative dans le cas particulier d'homologue à homologue.

On utilisera les notations générales suivantes :

HDR : C'est l'en-tête général de MIKEY, qui inclut les données en rapport avec le CSB de MIKEY (par exemple, CSB ID) et la transposition des informations dans le protocole de sécurité spécifique utilisé. Voir au paragraphe 6.1 la définition de la charge utile.

T : C'est l'horodatage, utilisé principalement pour empêcher les attaques en répétition. Voir au paragraphe 6.6 la définition de la charge utile et aussi au paragraphe 5.4 les autres informations en rapport avec l'horodatage.

IDx : C'est l'identité de l'entité x (IDi=Initiateur, IDr=Répondeur). Voir au paragraphe 6.7 la définition de la charge utile.

RAND : Chaîne d'octets aléatoire/pseudo-aléatoire, qui est toujours incluse dans le premier message de l'initiateur. RAND est utilisé comme une valeur de fraîcheur pour la génération de clé. Il n'est pas inclus dans les messages de mise à jour d'un CSB. Voir au paragraphe 6.11 la définition de la charge utile. Pour les recommandations d'aléa pour la sécurité, voir la [RFC1750].

SP : Ce sont les politiques de sécurité pour le protocole de sécurité des données. Voir au paragraphe 6.10 la définition de la charge utile.

3.1 Clé prépartagée

Dans cette méthode, la clé secrète prépartagée, s, est utilisée pour déduire le matériel de clé pour le chiffrement (encr_key) et pour la protection de l'intégrité (auth_key) des messages MIKEY, comme décrit au paragraphe 4.1.4. Les transformations de chiffrement et d'authentification sont décrites au paragraphe 4.2.

```
     Initiateur                                                  Répondeur
I_MESSAGE =  HDR, T, RAND, [IDi],[IDr], {SP}, KEMAC   -------->
                                                [<--------]   R_MESSAGE =  HDR, T, [IDr], V
```

Le principal objectif du message de l'initiateur (I_MESSAGE) est de transporter une ou plusieurs TGK (portées dans KEMAC) et un ensemble de paramètres de sécurité (SP) au répondeur d'une manière sûre. Comme le message de vérification du répondeur est facultatif, l'initiateur indique dans le HDR si il exige ou non un message de vérification de la part du répondeur.

KEMAC = E(encr_key, {TGK}) || MAC

La charge utile KEMAC contient un ensemble de sous charges utiles chiffrées et un MAC. Chaque sous charge utile comporte une TGK choisie au hasard et de façon indépendante par l'initiateur (et d'autres paramètres possibles en rapport, par exemple, la durée de vie de clé). Le MAC est un code d'authentification de message qui couvre la totalité du message MIKEY en utilisant la clé d'authentification, auth_key. Voir au paragraphe 6.2 la définition de la charge utile et au

paragraphe 5.2 une définition exacte du calcul du MAC.

Le principal objectif du message de vérification de la part du répondeur est d'obtenir l'authentification mutuelle. Le message de vérification, V, est un MAC calculé sur la totalité du message du répondeur, l'horodatage (le même que celui qui était inclus dans le message de l'initiateur), et les identités des deux parties, en utilisant la clé d'authentification. Voir aussi au paragraphe 5.2 la définition exacte du calcul du MAC de vérification et au paragraphe 6.9 la définition de la charge utile.

Le champ ID DEVRAIT être inclus, mais il PEUT être laissé en dehors quand on peut s'attendre à ce que l'homologue connaisse déjà l'identité de l'autre partie (autrement, il ne pourrait pas chercher la clé prépartagée). Par exemple, cela pourrait être le cas si l'ID est extrait de SIP.

Il est OBLIGATOIRE de mettre en œuvre cette méthode.

3.2 Chiffrement de clé publique

Initiateur **Répondeur**

```
I_MESSAGE =
 HDR, T, RAND, [IDi|CERTi], [IDr], {SP}, KEMAC, [CHASH], PKE, SIGNi ----->
                                                    [<-----]   R_MESSAGE =  HDR, T, [IDr], V
```

Comme dans le cas précédent, le principal objectif du message de l'initiateur est de transporter une ou plusieurs TGK et un ensemble de paramètres de sécurité au répondeur d'une manière sûre. On le fait en utilisant une approche d'enveloppe où les TGK sont chiffrées (et protégées en intégrité) avec des clés déduites d'une "clé enveloppe" choisie de façon aléatoire ou pseudo-aléatoire. La clé enveloppe est envoyée au répondeur chiffrée avec la clé publique du répondeur.

La PKE contient la clé enveloppe chiffrée : PKE = E(PKr, env_key). Elle est chiffrée en utilisant la clé publique du répondeur (PKr). Si le répondeur possède plusieurs clés publiques, l'initiateur peut indiquer la clé utilisée dans la charge utile CHASH (voir au paragraphe 6.8).

Le KEMAC contient un ensemble de sous charges utiles chiffrées et un MAC :

KEMAC = E(encr_key, IDi || {TGK}) || MAC

La première charge utile (IDi) dans KEMAC est l'identité de l'initiateur (ce n'est pas un certificat, mais généralement le même ID que celui spécifié dans le certificat). Chacune des charges utiles suivantes (TGK) comporte un TGK choisi au hasard et indépendamment par l'initiateur (et éventuellement d'autres paramètres en rapport, par exemple, la durée de vie de clé). La partie chiffrée est alors suivie par un MAC, qui est calculé sur la charge utile KEMAC. La encr_key et la auth_key sont déduites de la clé enveloppe, env_key, comme spécifié au paragraphe 4.1.4. Voir aussi au paragraphe 6.2 la définition de charge utile.

SIGNi est une signature qui couvre le message MIKEY entier, utilisant la clé de signature de l'initiateur (voir aussi au paragraphe 5.2 la définition exacte).

Le principal objectif du message de vérification provenant du répondeur est d'obtenir l'authentification mutuelle. Comme le message de vérification V provenant du répondeur est facultatif, l'initiateur indique dans le HDR si il exige ou non un message de vérification de la part du répondeur. V est calculé de la même façon que dans le mode de clé prépartagée (voir aussi au paragraphe 5.2 la définition exacte). Voir au paragraphe 6.9 la définition de la charge utile.

Noter qu'il y aura un IDi chiffré et éventuellement aussi un IDi non chiffré. Celui qui est chiffré est utilisé avec le MAC comme une contre mesure pour certaines attaques par interposition, tandis que celui qui n'est pas chiffré est toujours utile pour que le répondeur identifie immédiatement l'initiateur. L'IDi chiffré DOIT toujours être vérifié comme étant égal à l'IDi attendu.

Il est possible de mettre en antémémoire la clé enveloppe, de sorte qu'elle puisse être utilisée comme une clé prépartagée. Il n'est pas recommandé que cette clé soit indéfiniment en antémémoire (cependant il relève de la politique locale d'en décider). Cette fonction peut être très pratique durant la durée de vie d'un CSB, si une nouvelle session de chiffrement doit être ajoutée (ou si une qui a expira est retirée). Alors, la clé prépartagée peut être utilisée, au lieu des clés publiques (voir aussi au paragraphe 4.5). Si l'initiateur indique que la clé enveloppe devrait être mise en antémémoire, la clé doit au moins être mise en antémémoire durant la durée de vie entière du CSB.

Les champs d'ID en clair et le certificat DEVRAIENT être inclus, mais ils PEUVENT être laissés en dehors lorsque on

peut s'attendre à ce que l'homologue sache déjà l'identité de l'autre partie, ou qu'on peut obtenir le certificat d'une autre manière. Par exemple, ce pourrait être le cas si l'ID est extrait de SIP.

Pour le traitement du certificat, de l'autorisation, et des politiques, voir au paragraphe 4.3.

Il est OBLIGATOIRE de mettre en œuvre cette méthode.

3.3 Échange de clé Diffie-Hellman

Pour un groupe cyclique fixé (G,*) ayant fait l'objet d'un accord, on va noter g un générateur pour ce groupe. Les choix des paramètres sont donnés au paragraphe 4.2.7. Les autres transformations ci-dessous sont décrites au paragraphe 4.2.

Cette méthode crée une clé DH, qui est utilisée comme TGK. Cette méthode ne peut pas être utilisée pour créer des clés de groupe ; elle ne peut être utilisée que pour créer des clés pour un seul couple d'homologues. La mise en œuvre de cette méthode est FACULTATIVE.

```
     Initiateur                                                    Répondeur
I_MESSAGE = HDR, T, RAND, [IDi|CERTi],[IDr] {SP}, DHi, SIGNi  ---------->
                                          <--------- R_MESSAGE =  HDR, T, [IDr|CERTr], IDi, DHr, DHi, SIGNr
```

Le principal objectif du message de l'initiateur est de fournir, de façon sûre au répondeur sa valeur DH (DHi) g^(xi), où xi DOIT être choisi de façon aléatoire/pseudo-aléatoire et secrète, et un ensemble de paramètres de protocole de sécurité.

SIGNi est une signature couvrant le message MIKEY de l'initiateur, I_MESSAGE, en utilisant la clé de signature de l'initiateur (voir la définition exacte au paragraphe 5.2).

Le principal objectif du message du répondeur est de fournir d'une façon sûre à l'initiateur la valeur du répondeur (DHr) g^(xr), où xr DOIT être choisi de façon aléatoire/pseudo-aléatoire et secrète. L'horodatage qui est inclus dans la réponse est la même que celle incluse dans le message de l'initiateur.

SIGNr est une signature qui couvre le message MIKEY du répondeur, R_MESSAGE, en utilisant la clé de signature du répondeur (voir la définition exacte au paragraphe 5.2).

Les paramètres de groupe DH (par exemple, le groupe G, le générateur g) sont choisis par l'initiateur et signalés au répondeur. Les deux parties calculent la TGK, g^(xi*xr) à partir des valeurs DH échangées.

Noter que cette approche n'exige pas que l'initiateur doive posséder un des certificats du répondeur avant l'établissement. Il est en fait suffisant que le répondeur inclue son certificat signant dans la réponse.

Les champs ID et le certificat DEVRAIENT être inclus, mais ils PEUVENT être laissés en dehors lorsque on peut s'attendre à ce que l'homologue sache déjà l'ID de l'autre partie (ou qu'il peut obtenir le certificat de quelque autre manière). Par exemple, ce pourrait être le cas si l'ID est extrait de SIP.

Pour le traitement du certificat, de l'autorisation, et des politiques, voir le paragraphe 4.3.

4. Fonctions de gestion de clé choisies

MIKEY gère les clés symétriques de deux façons principales. Premièrement, à la suite du transport de clé ou de l'échange de clés de TGK (et autres paramètres) comme défini par une des trois méthodes ci-dessus, MIKEY entretient une transposition entre les identifiants de SA de données et les SA de données, où les identifiants utilisés dépendent du protocole de sécurité en question, voir au paragraphe 4.4. Donc, lorsque le protocole de sécurité demande une SA de données, connaissant un tel identifiant de SA de données, une SA de données mise à jour sera obtenue. En particulier, un matériel de clés correct, une ou des TEK, peuvent devoir être déduits. La déduction de la ou des TEK (et autre matériel de clé) est faite à partir d'une TGK et est décrite au paragraphe 4.1.3.

Deuxièmement, pour utilisation au sein de MIKEY lui-même, deux procédures de gestion de clé sont nécessaires :

* dans le cas clé prépartagée, la déduction du chiffrement et du matériel de clé d'authentification à partir d'une seule clé prépartagée, et
* dans le cas de la clé publique, la déduction d'un matériel de clé similaire à partir de la clé d'enveloppe transportée.

Ces deux méthodes de déduction de clé sont spécifiées au paragraphe 4.1.4.

Toutes les fonctions de déduction de clé mentionnées ci-dessus se fondent sur une fonction pseudo aléatoire, définie ci-dessous.

4.1 Calcul de clé

Dans ce qui suit, on définit une méthode générale (fonction pseudo aléatoire) pour déduire une ou plusieurs clés d'une clé "maîtresse". Cette méthode est utilisée pour déduire :

* les TEK d'une TGK et de la valeur RAND,
* la clé de chiffrement, d'authentification, ou de salage à partir d'une clé enveloppe prépartagée et de la valeur RAND.

4.1.1 Hypothèses

On suppose que les paramètres suivants sont en place :

csb_id : identifiant de faisceau de session de chiffrement (entier non signé de 32 bits)
cs_id : l'identifiant de session de chiffrement (entier non signé de 8 bits)
RAND : chaîne binaire (pseudo-)aléatoire de (au moins) 128 bits envoyée par l'initiateur dans l'échange initial.

La méthode de déduction de clé a les paramètres d'entrée suivants :inkey : clé d'entrée de la fonction de déduction
inkey_len : longueur en bits de la clé d'entrée
label : étiquette spécifique, qui dépend du type de la clé à déduire, du RAND, et des identifiants de session
outkey_len : longueur désirée en bits de la clé de sortie.

La méthode de déduction de clé a la sortie suivante :
outkey : la clé de sortie de la longueur désirée.

4.1.2 Description de la PRF par défaut

Soit HMAC la fonction d'authentification de message fondée sur SHA-1, voir la [RFC2104], et [SHA-1]. De même que dans la [RFC2246], on définit :

P (s, label, m) = HMAC (s, A_1 || label) || HMAC (s, A_2 || label) || ... HMAC (s, A_m || label)

où
A_0 = label *(étiquette)*,
A_i = HMAC (s, A_(i-1))
s est une clé (définie ci-dessous)
m est un entier positif (aussi défini ci-dessous).

Les valeurs de label dépendent du cas dans lequel la PRF est invoquée, et les valeurs sont spécifiées dans ce qui suit pour la PRF par défaut. Donc, on notera que d'autres PRF ajoutées ultérieurement à MIKEY PEUVENT spécifier des paramètres d'entrée différents.

La procédure qui suit décrit une fonction pseudo aléatoire, notée PRF(inkey,label), fondée sur la fonction P ci-dessus, appliquée pour calculer la clé de sortie, outkey :

* soit n = inkey_len / 256, arrondi à l'entier le plus proche si ce n'est déjà un entier
* partager la inkey en n blocs, inkey = s_1 || ... || s_n, où * tous les s_i, excepté éventuellement s_n, font 256 bits chacun
* soit m = outkey_len / 160, arrondi à l'entier le plus proche, si ce n'est déjà un entier.

(Les valeurs "256" et "160" sont égales, respectivement, à la moitié de la taille du bloc d'entrée et à la taille du hachage de sortie entier, du hachage SHA-1 au titre de la fonction P.)

Puis, la clé de sortie, outkey, est obtenue comme les outkey_len bits de plus fort poids de

PRF(inkey, label) = P(s_1, label, m) OUX P(s_2, label, m) OUX ... OUX P(s_n, label, m).

4.1.3 Génération des clés d'une TGK

Dans ce qui suit, on décrit comment le matériel de clé est déduit d'une TGK, en supposant donc qu'une transposition de l'identifiant de la SA de données en la TGK correcte a déjà été faite, conformément au paragraphe 4.4.

La méthode de déduction de clé DEVRA être exécutée en utilisant la PRF ci-dessus avec les paramètres d'entrée suivants :
inkey : TGK
inkey_len : longueur en bits de la TGK
label : constante || cs_id || csb_id || RAND
outkey_len : longueur en bits de la clé de sortie.

La partie constante de l'étiquette dépend du type de clé qui est à générer. La constante 0x2AD01C64 est utilisée pour générer une TEK à partir de la TGK. Si le protocole de sécurité lui-même ne prend pas en charge la déduction de clé pour l'authentification et le chiffrement à partir de la TEK, des clés d'authentification et de chiffrement distinctes PEUVENT être créées directement pour le protocole de sécurité en remplaçant respectivement 0x2AD01C64 par 0x1B5C7973 et 0x15798CEF , et outkey_len par la ou les longueurs de clé désirées dans chaque cas.

Une clé de salage peut aussi être déduite de la TGK en utilisant la constante 0x39A2C14B. Noter que la sous charge utile Données de clé (paragraphe 6.13) peut porter un sel. Le protocole de sécurité qui a besoin d'une clé de sel DEVRA utiliser la clé de sel portée dans la sous charge utile Données de clé (dans les cas de clé prépartagée et de clé publique) lorsque elle est présente. Si cela n'est pas envoyé, il est alors possible de déduire la clé de sel via la fonction de déduction de clé décrite ci-dessus.

Le tableau ci-dessous résume les valeurs constantes utilisées pour générer les clés à partir d'une TGK.

Constante	**Clé déduite de la TGK**
0x2AD01C64	TEK
0x1B5C7973	clé d'authentification
0x15798CEF	clé de chiffrement
0x39A2C14B	clé de salage

Tableau 4.1.3 : Valeurs de constantes pour la déduction des clés d'une TGK

Noter que ces valeurs constantes de 32 bits (énumérées dans le tableau ci-dessus) sont tirées des chiffres décimaux de e (c'est-à-dire, 2,7182...), où chaque constante consiste en neuf chiffres décimaux (par exemple, les neuf premiers chiffres décimaux 718281828 = 0x2AD01C64). La chaîne de neuf chiffres décimaux n'est pas choisie au hasard, mais comme "tronçons" consécutifs des décimales de e.

4.1.4 Génération des clés pour les messages MIKEY à partir d'une clé d'enveloppe/prépartagée

Cette déduction est pour former une clé de chiffrement symétrique (et la clé de salage) pour le chiffrement de la TGK dans les méthodes de clé prépartagée et de clé publique. C'est aussi utilisé pour déduire la clé symétrique utilisée pour le code d'authentification de message (MAC, *message authentication code*) dans ces messages, et les messages de vérification correspondants. Donc, cette déduction est nécessaire afin d'obtenir des clés différentes pour le chiffrement et pour le MAC (et dans le cas de clé prépartagée, il va en résulter un matériel de clé frais pour chaque nouveau CSB). Les paramètres pour la PRF par défaut sont ici :
inkey : la clé enveloppe ou la clé prépartagée
inkey_len : longueur en bits de inkey
label : constante || 0xFF || csb_id || RAND
outkey_len : longueur désirée en bits de la clé de sortie

La partie constante de l'étiquette dépend du type de clé qui est à générer à partir d'une clé d'enveloppe/prépartagée, comme résumé ci-dessous.

Constante	**Clé déduite**
0x150533E1	clé de chiffrement
0x2D22AC75	clé d'authentification
0x29B88916	clé de salage

Tableau 4.1.4 : Valeurs de constantes pour la déduction des clés d'une clé d'enveloppe/prépartagée

4.2 Transformations prédéfinies et formats d'horodatage

Ce paragraphe identifie les transformations par défaut pour MIKEY. Il est obligatoire de mettre en œuvre et prendre en charge les transformations suivantes dans le cas respectivement concerné. De nouvelles transformations pourront être ajoutées à l'avenir (voir d'autres lignes directrices au paragraphe 4.2.9).

4.2.1 Fonctions de hachage

Dans MIKEY, il est OBLIGATOIRE de mettre en œuvre SHA-1 comme fonction de hachage par défaut.

4.2.2 Générateur de nombres pseudo-aléatoires et de la PRF

Un générateur de nombres aléatoires ou pseudo aléatoires cryptographiquement sûr DOIT être utilisé pour la génération du matériel de clés et des noms occasionnels, par exemple, [BMGL]. Cependant, le choix de celui qui est à utiliser est spécifique de la mise en œuvre (car ce choix ne va pas affecter l'interopérabilité).

Pour les déductions de clés, il est OBLIGATOIRE de mettre en œuvre la PRF spécifiée au paragraphe 4.1. D'autres PRF PEUVENT être ajoutées par des RFC en cours de normalisation qui spécifient les constructions de PRF et leur exact utilisation avec MIKEY.

4.2.3 Chiffrement du transport de donnés de clé

Le chiffrement de transport de clé par défaut et de mise en œuvre obligatoire est AES en mode compteur, comme défini dans la [RFC3711], en utilisant une clé de 128 bits déduite comme décrit au paragraphe 4.1.4, SRTP_PREFIX_LENGTH réglé à zéro, et en utilisant le vecteur d'initialisation

IV = (S OUX (0x0000 || CSB ID || T)) || 0x0000,

où S est une clé de salage de 112 bits, déduite aussi comme au paragraphe 4.1.4, et où T est l'horodatage de 64 bits envoyé par l'initiateur.

Note : Cela restreint la taille maximum qui peut être chiffrée à 2^23 bits, ce qui est assez pour tous les cas pratiques [RFC3711].

L'algorithme de chiffrement NUL (c'est-à-dire, pas de chiffrement) peut être utilisé (mais sa mise en œuvre est FACULTATIVE). Noter qu'il NE DOIT PAS être utilisé si les protocole sous-jacents ne peuvent pas garantir la sécurité. La principale raison de son inclusion tient à des scénarios spécifiques de SIP, où SDP est protégé de bout en bout. Pour ce scénario, MIKEY PEUT être utilisé avec la méthode des clés prépartagées, le chiffrement NUL, et l'algorithme d'authentification NUL (voir au paragraphe 4.2.4) tout en s'appuyant sur la sécurité de SIP. Utiliser cette option avec prudence !

La fonction AES d'enveloppe de clé de la [RFC3394] est incluse comme méthode FACULTATIVE de mise en œuvre. Si la fonction d'enveloppe de clé est utilisée dans la méthode de clé publique, il est RECOMMANDÉ d'utiliser le MAC NUL, car l'enveloppe de clé elle-même va assurer la protection de l'intégrité du contenu chiffré (noter cependant que le MAC NUL NE DEVRAIT PAS être utilisé dans le cas des clés prépartagées, car dans ce cas le MAC couvre le message entier). La clé de 128 bits et un sel de 64 bits, S, sont déduits selon le paragraphe 4.1.4 et l'IV d'enveloppe de clé est alors réglé à S.

4.2.4 MAC et fonction de vérification de message

MIKEY utilise une étiquette d'authentification de 160 bits, générée par HMAC avec SHA-1 comme méthode OBLIGATOIRE de mise en œuvre, voir la [RFC2104]. Les clés d'authentification sont déduites conformément au paragraphe 4.1.4. Noter que la taille de clé d'authentification DEVRAIT être égale à celle du résultat de la fonction de hachage (par exemple, pour HMAC-SHA-1, une clé d'authentification de 160 bits est utilisée) [RFC2104].

L'algorithme d'authentification NUL (c'est-à-dire, aucun MAC) peut être utilisée avec l'algorithme de chiffrement NUL (mais sa mise en œuvre est FACULTATIVE). Noter que cela NE DOIT PAS être utilisé sauf si les protocoles sous-jacents peuvent garantir la sécurité. La principale raison de son inclusion est pour des scénarios spécifiques de SIP, où SDP est protégé de bout en bout. Pour ce scénario, MIKEY PEUT être utilisé avec la méthode des clés prépartagées et le chiffrement et l'algorithme d'authentification NUL, tout en s'appuyant sur la sécurité de SIP. Utiliser cette option avec précaution !

4.2.5 Chiffrement de clé d'enveloppe

L'algorithme de chiffrement à clé publique appliqué est défini par, et dépend, du certificat utilisé. Il est OBLIGATOIRE de prendre encharge RSA PKCS n°1, v1.5, et il est RECOMMANDÉ de prendre aussi en charge RSA OAEP [PSS].

4.2.6 Signatures numériques

L'algorithme de signature appliqué est défini par, et dépend du certificat utilisé. Il est OBLIGATOIRE de prendre en charge RSA PKCS n°1, v1.5, et il est RECOMMANDÉ de prendre aussi en charge RSA PSS [PSS].

4.2.7 Groupes Diffie-Hellman

L'échange de clé Diffie-Hellman, lorsque il est pris en charge, utilise OAKLEY 5 [RFC2412] comme mise en œuvre obligatoire. OAKLEY 1 et OAKLEY 2 PEUVENT tous deux être utilisés (mais ce sont des mises en œuvre FACULTATIVES).

Voir au paragraphe 4.2.9 les lignes directrices sur la spécification d'un nouveau groupe DH à utiliser au sein de MIKEY.

4.2.8 Horodatages

L'horodatage est comme défini dans NTP [RFC1305], c'est-à-dire, un nombre de 64 bits en secondes par rapport à 0 h le 1 er janvier 1900. Une mise en œuvre DOIT connaître (et prendre en compte) le fait que le compteur va déborder approximativement tous les 136 ans. Il est RECOMMANDÉ que l'heure soit toujours spécifiée en UTC.

4.2.9 Ajout de nouveaux paramètres à MIKEY

Il y a deux jeux de paramètres différents qui peuvent être ajoutés à MIKEY. Le premier est un jeu de transformations de MIKEY (nécessaires pour l'échange lui-même) et le second est les SA de données.

De nouvelles transformations et paramètres (incluant de nouvelles politiques) DEVRONT être ajoutées en enregistrant auprès de l'IANA (selon la [RFC2434], voir aussi la Section 10) un nouveau numéro pour la charge utile concernée, et aussi si nécessaire, en documentant comment est utilisé le nouveau paramètre/transformation. Il peut parfois être suffisant de pointer sur un document déjà spécifié pour l'usage, par exemple, lorsque on ajoute une nouvelle fonction de hachage, déjà normalisée.

Dans le cas de l'ajout d'un nouveau groupe DH, le groupe DOIT être spécifié dans une RFC en cours de normalisation (il est RECOMMANDÉ que le groupe spécifié utilise le même format qu'utilisé dans la [RFC2412]). Un numéro peut alors être alloué par l'IANA pour qu'un tel groupe soit utilisé dans MIKEY.

Pour ajouter la prise en charge d'un nouveau protocole de sécurité des données, on DOIT spécifier ce qui suit :

* Une sous charge utile transposée (voir au paragraphe 6.1). C'est utilisé pour permettre de transposer une session de chiffrement en la bonne instance de protocole de sécurité des données et éventuellement aussi de fournir les paramètres individuels pour chaque protocole de sécurité de données.
* Une charge utile de politique, c'est-à-dire, la spécification des paramètres et des valeurs prises en charge.
* Les lignes directrices générales d'usage.

4.3. Certificats, politiques et autorisation

4.3.1 Traitement des certificat

Le traitement des **c**ertificats peut impliquer un certain nombre de tâches supplémentaires qu'on ne montre pas ici, et affecter l'inclusion de certaines parties du message (cf. [RFC3280]). Cependant, on peut faire les observations suivantes :

* L'initiateur doit normalement trouver le certificat du répondeur afin d'envoyer le premier message. Si l'initiateur n'a pas déjà le certificat du répondeur, cela peut impliquer un ou plusieurs allers-retours à un agent de répertoire central.
* Il serait possible que l'initiateur omette son propre certificat et se contente que le répondeur obtienne ce certificat par d'autres moyens. Cependant, on recommande de ne faire cela que lorsque il est raisonnable de s'attendre à ce que le répondeur ait mis le certificat en antémémoire à partir d'une connexion antérieure. Accéder au certificat signifierait autrement des allers-retours supplémentaires également pour le répondeur.
* La vérification des certificats en utilisant des listes de révocation de certificat (CRL, *Certificate Revocation List*) [RFC3280] ou des protocoles tels que OCSP [RFC2560] peut être nécessaire. Toutes les parties à un échange MIKEY devraient avoir une politique locale qui indique si de telles vérifications sont faites, comment elles sont faites, et à quelle fréquence. Noter qu'effectuer les vérifications peut impliquer des échanges de messages supplémentaires.

4.3.2 Autorisation

Il y a en général deux modèles différents pour prendre les décisions d'autorisation pour l'initiateur et pour le répondeur, dans le contexte des applications ciblées par MIKEY :

* Configuration spécifique d'homologue à homologue. L'usager a configuré l'application pour faire confiance à un homologue spécifique.
  Lorsque on utilise des secrets prépartagés, c'est le seul schéma disponible. Normalement, l'entrée de configuration du secret prépartagé sert à signifier que l'autorisation est implicite.
  Dans certains cas, on pourrait aussi utiliser cela avec des clés publiques, par exemple, si deux homologues échangent des clés hors ligne et les configurent pour les utiliser à faire fonctionner MIKEY.
* Racine de confiance. L'usager accepte tous les homologues qui prouvent qu'ils ont un certificat produit par un CA spécifique. La granularité des décisions d'autorisation n'est pas très précise dans cette méthode. Pour rendre cette méthode possible, tous les participants au protocole MIKEY ont besoin de configurer une ou plusieurs racines de confiance. Les participants doivent aussi être capables d'effectuer la validation de la chaîne de certificats, et éventuellement de transférer plus d'un seul certificat dans les messages MIKEY (voir aussi au paragraphe 6.7).

En pratique, une combinaison des deux méthodes mentionnées pourrait être avantageuse. Aussi, la possibilité qu'un usager exclue explicitement un homologue spécifique (ou un sous arbre) dans une chaîne de confiance pourrait être nécessaire.

Ces politiques d'autorisation visent les scénarios MIKEY a à c du paragraphe 2.1, où l'initiateur agit comme propriétaire du groupe et est aussi le seul qui puisse en inviter d'autres. Cela implique que pour chaque répondeur, les clés distribuées NE DOIVENT PAS être redistribuées aux autres parties.
Dans une situation de beaucoup à beaucoup, lorsque les fonctions de contrôle du groupe sont réparties (et/ou lorsque il est possible de déléguer les fonctions de contrôle du groupe aux autres) un moyen de distribuer les informations d'autorisation sur qui peut être ajouté au groupe DOIT exister. Cependant, il sort du domaine d'application du présent document de spécifier comment cela devrait être fait.
Pour toute situation de communication plus large, une infrastructure d'autorisation externe peut être utilisée (suivant les hypothèses de la [RFC4046]).

4.3.3 Politiques de données

Incluses dans l'échange de messages sont transmises les politiques (c'est-à-dire, les paramètres de sécurité) pour le protocole de sécurité des données. Les politiques sont définies dans une charge utile distincte et sont spécifiques du protocole de sécurité (voir aussi le paragraphe 6.10). Avec les clés, la période de validité de ces politiques peut aussi être spécifiée. Par exemple, cela peut être fait avec un SPI (ou MKI SRTP) ou avec un intervalle (par exemple, un intervalle de numéro de séquence pour SRTP) selon le protocole de sécurité.
De nouveaux paramètres peuvent être ajoutés à une politique en documentant comment ils devraient être interprétés par MIKEY et aussi en enregistrant de nouvelles valeurs dans l'espace de noms approprié de l'IANA. Si une politique complètement nouvelle est nécessaire, voir les lignes directrices au paragraphe 4.2.9.

4.4 Restitution de SA de données

La restitution d'une SA de données va dépendre du protocole de sécurité, car des protocoles de sécurité différents vont avoir des caractéristiques différentes. Lorsque on ajoute la prise en charge d'un protocole de sécurité à MIKEY, une interface de la façon dont le protocole de sécurité restitue la SA de données de MIKEY DOIT être spécifiée (avec les politiques qui peuvent être négociées).

Pour SRTP, le SSRC (voir la [RFC3711]) est un des paramètres utilisés pour restituer la SA de données (alors que MKI peut être utilisé pour indiquer la TGK/TEK utilisée pour la SA de données). Cependant, le SSRC n'est pas suffisant. Pour la restitution de la SA de données à partir de MIKEY, il est RECOMMANDÉ que la mise en œuvre de MIKEY prenne en charge une recherche utilisant l'adresse réseau et l'accès de destination avec le SSRC. Noter que MIKEY n'envoie pas d'adresses réseau ou d'accès. Une des raisons en est que cela ne peut pas être connu à l'avance. Aussi, si il existe un NAT sur le parcours, il peut y avoir des problèmes. Lorsque on utilise SIP ou RTSP, la vue locale de l'adresse et de l'accès de destination peut être obtenue de SIP ou de RTSP. MIKEY peut alors utiliser ces adresses comme indice pour la recherche de SA de données.

4.5 Changement de clé TGK et mise à jour de CSB

MIKEY donne un moyen pour mettre à jour le CSB (par exemple, en transportant une nouvelle TGK/TEK ou en ajoutant une nouvelle session de chiffrement au CSB). La mise à jour du CSB est faite en exécutant à nouveau MIKEY, par exemple, avant l'arrivée à expiration d'une TEK, ou lorsque une nouvelle session de chiffrement est ajoutée au CSB. Noter que MIKEY ne fournit pas le changement de clés dans le sens de GKMARCH, mettant seulement les clés à jour par les messages en envoi individuel normaux.

Lorsque MIKEY est exécuté à nouveau pour mettre à jour le CSB, il n'est pas nécessaire d'inclure les certificats et les autres informations qui avaient été fournies dans le premier échange, par exemple, toutes les charges utiles qui sont

statiques ou facultatives peuvent être laissées de côté (voir la Figure 4.1).
Le nouvel échange de messages DOIT utiliser le même identifiant de CSB que l'échange initial, mais DOIT utiliser un nouvel horodatage. Un nouveau RAND NE DOIT PAS être inclus dans l'échange de message (le RAND n'aura d'effet que dans l'échange initial). Si on le désire, de nouvelles sessions de chiffrement sont ajoutées dans le message de mise à jour. Noter qu'un message de mise à jour MIKEY n'a pas besoin de contenir de nouveau matériel de clé (par exemple, une nouvelle TGK). Dans ce cas, la session de chiffrement continue d'utiliser le matériel de clé établi précédemment, tout en mettant à jour les nouvelles informations.
Comme expliqué au paragraphe 3.2, la clé enveloppe peut être "mise en antémémoire" comme une clé prépartagée (cela est indiqué par l'initiateur dans le premier message envoyé). S'il en est ainsi, le message de mise à jour est un message de clé prépartagée avec la clé enveloppe mise en antémémoire comme clé prépartagée ; il NE DOIT PAS être un message de clé publique. Si le message de clé publique est utilisé mais que la clé enveloppe n'est pas mise en antémémoire, l'initiateur DOIT fournir une nouvelle clé enveloppe chiffrée qui puisse être utilisée dans le message de vérification. Cependant, l'initiateur n'a pas besoin de fournir d'autres clés.

La Figure 4.1 permet de visualiser les messages de mise à jour qui peuvent être envoyés, y compris les parties facultatives. La principale différence avec le message original est qu'il est facultatif d'inclure les TGK (ou valeurs DH dans la méthode DH). Voir aussi à la Section 3 les détails sur les méthodes spécifiques.

Par définition, un CSB peut contenir plusieurs CS. Le problème qui peut se poser est celui de synchroniser le changement de TGK si un SPI (ou une fonctionnalité similaire, par exemple, MKI dans la [RFC3711]) n'est pas utilisé. Il est donc RECOMMANDÉ qu'un SPI ou MKI soit utilisé, si plus d'un CS est présent.

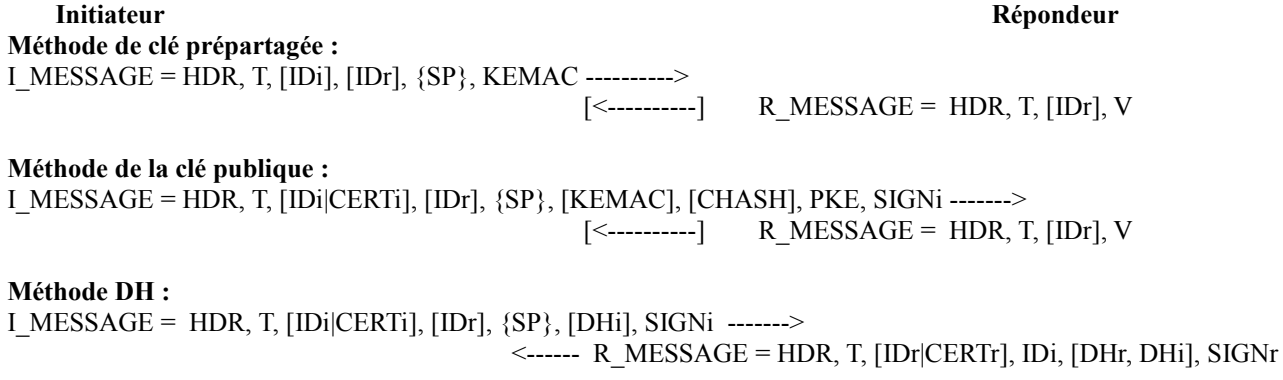


Figure 4.1 : Messages de mise à jour

Noter que pour la méthode DH, si l'initiateur inclut la charge utile DHi, le répondeur DOIT alors inclure DHr et DHi. Si l'initiateur n'inclut pas DHi, le répondeur NE DOIT PAS inclure DHr ou DHi.

5. Comportement et traitement de message

Chaque message envoyé par l'initiateur ou le répondeur est constitué par un ensemble de charges utiles. La présente section décrit comment sont créés les messages et aussi quand ils peuvent être utilisés.

5.1 Généralités

5.1.1 Découverte des capacités

L'initiateur indique la politique de sécurité à utiliser (c'est-à-dire, en termes d'algorithmes de protocole de sécurité). Si le répondeur ne la prend pas en charge (pour une raison quelconque) le répondeur peut, en plus d'un message d'erreur (qui indiquera qu'il ne prend pas en charge les paramètres) renvoyer ses propres capacités (négociation) pour permettre à l'initiateur de choisir un ensemble commun de paramètres. Cela se fait en incluant une ou plusieurs charges utiles de politique de sécurité dans le message d'erreur envoyé en réponse (voir au paragraphe 5.1.2.). Plusieurs attributs peuvent être fournis à la suite dans la réponse. On fait cela pour réduire autant que possible le nombre d'allers-retours (c'est-à-dire, dans la plupart des cas, lorsque la politique est acceptée dès la première fois, un aller-retour est suffisant). Si le répondeur n'accepte pas l'offre, l'initiateur doit produire un nouveau message MIKEY.

Si le répondeur ne veut pas (ou n'est pas capable de) fournir la sécurité, ou si les parties ne peuvent simplement pas se mettre d'accord, il appartient aux politiques des parties de décider du comportement à tenir, par exemple, en acceptant ou

rejetant une communication non sûre.

Noter qu'il n'entre pas dans les intentions de ce protocole d'avoir une large variété d'options, car on suppose qu'il devrait rarement se produire qu'une offre soit refusée.

Dans les scénarios de un à plusieurs et de beaucoup à beaucoup qui utilisent la communication en diffusion groupée, un des problèmes est bien sûr qu'il DOIT y avoir une politique de sécurité commune pour tous les receveurs. Cela limite les possibilités de négociation.

5.1.2 Traitement d'erreur

Du fait du protocole de gestion de clés, toutes les erreurs DEVRAIENT être rapportées à ou aux homologues par un message d'erreur. L'initiateur DEVRAIT donc toujours être prêt à recevoir un tel message du répondeur.

Si le répondeur ne prend pas en charge l'ensemble de paramètres suggéré par l'initiateur, le message d'erreur DEVRAIT comporter les paramètres pris en charge (voir aussi au paragraphe 5.1.1).

Le format du message d'erreur est le suivant :

HDR, T, {ERR}, {SP}, [V|SIGNr]

Noter que si la défaillance est due à l'incapacité à authentifier l'homologue, le message d'erreur est FACULTATIF et n'a pas besoin d'être authentifié. Il appartient à la politique locale de déterminer comment traiter cette sorte de message. Cependant, si en réponse à un échec d'authentification un message d'erreur signé est retourné, cela peut être utilisé à des fins de déni de service (contre le répondeur). De même, un message d'erreur non authentifié pourrait être envoyé à l'initiateur afin de le conduire à supprimer le CSB. Il est fortement RECOMMANDÉ que la politique locale prenne cela en considération. Donc, en cas d'échec d'authentification, une recommandation serait de ne pas authentifier un tel message d'erreur, et que lorsque un message d'erreur non authentifié est reçu, de ne le voir que comme une indication de ce qui pourrait aller de travers.

5.2 Création d'un message

Pour créer un message MIKEY, on crée d'abord une charge utile d'en-tête commun. Cette charge utile est alors suivie, selon le type de message, par un ensemble de charges utiles d'informations (par exemple, charge utile de valeur DH, de signature, de politique de sécurité). Les charges utiles définies et le codage exact de chaque charge utile sont décrits à la Section 6.

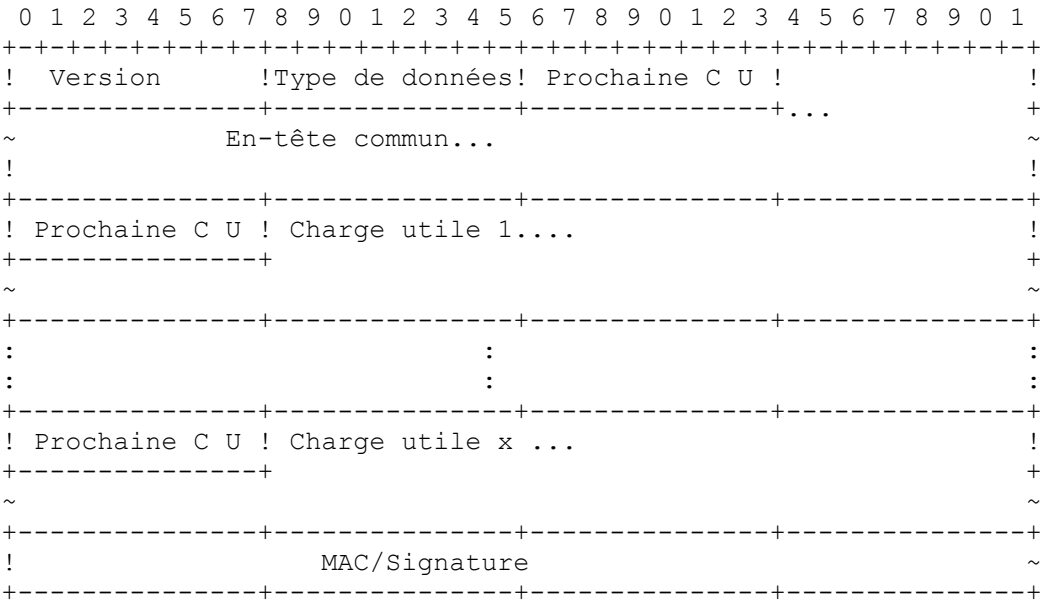


Figure 5.1 : Exemple de message de charge utile MIKEY

Noter que les charges utiles sont alignées sur l'octet et non sur 32 bits.

Le processus de génération d'un message MIKEY comporte les étapes suivantes :

* Créer un message initial MIKEY commençant par la charge utile En-tête commun.
* Enchaîner les charges utiles nécessaires du message MIKEY (voir les définitions d'échange pour les charges utiles qui peuvent être incluses, et l'ordre recommandé).
* À la dernière étape (pour les messages qui doivent être authentifiés, cela comporte aussi le message de vérification) créer et enchaîner la charge utile MAC/signature sans remplir le champ MAC/signature (si un champ Prochaine charge utile est inclus dans cette charge utile, elle est réglée à Dernière charge utile).
* Calculer le MAC/signature sur le message MIKEY entier, sauf le champ MAC/Signature, et ajouter le MAC/signature dans le champ. En cas de message de vérification, le Identity_i || Identity_r || Horodatage DOIT suivre directement le message MIKEY dans le calcul Vérification MAC. Note que les identités et horodatages ajoutés sont identiques à ceux transportés dans les charges utiles ID et T.

Dans le cas de clé publique, la charge utile Transport de données de clé est générée par l'enchaînement de l'IDi avec les TGK. Cela est ensuite chiffré et placé dans le champ Données. Le MAC est calculé sur la charge utile Transport de données de clé entière sauf le champ MAC. Avant de calculer le MAC, le champ Prochaine charge est réglé à zéro.

Noter que tous les messages provenant de l'initiateur DOIVENT utiliser un horodatage unique. Le répondeur ne crée par un nouvel horodatage, mais utilise l'horodatage utilisé par l'initiateur.

5.3 Analyse de message

En général, l'analyse d'un message MIKEY est faite en extrayant charge utile après charge utile et en vérifiant qu'il n'y a pas d'erreur. La procédure exacte est spécifique de la mise en œuvre ; cependant, pour le répondeur, il est RECOMMANDÉ de suivre la procédure suivante :

* Extraire l'horodatage et vérifier si il est dans le biais d'horloge admissible (si non, éliminer le message). Vérifier aussi l'antémémoire de répétition (paragraphe 5.4) afin que le message ne soit pas répété . Si le message est répété, l'éliminer.
* Extraire l'ID et l'algorithme d'authentification (si il n'est pas inclus, supposer l'algorithme par défaut).
* Vérifier le MAC/signature.
* Si l'authentification ne réussit pas, un message d'erreur Échec d'authentification PEUT être envoyé à l'initiateur. Le message est ensuite éliminé sans autre traitement. Voir aussi au paragraphe 5.1.2 le traitement des erreurs.
* Si l'authentification réussit, le message est traité et aussi ajouté à l'antémémoire de répétition ; le traitement est spécifique de la mise en œuvre. Noter aussi que seuls les messages bien authentifiés sont mémorisés dans l'antémémoire de répétition.
* Si des paramètres non pris en charge ou des erreurs surviennent durant le traitement, ils PEUVENT faire l'objet d'un rapport à l'initiateur par l'envoi d'un message d'erreur. Le traitement est alors interrompu. Le message d'erreur peut aussi comporter des charges utiles pour décrire les paramètres pris en charge.
* Si le traitement a réussi et dans le cas où l'initiateur l'a demandé, un message de vérification/réponse PEUT être créé et envoyé à l'initiateur.

5.4 Traitement des répétitions et usage de l'horodatage

MIKEY n'utilise pas de mécanisme de défi-réponse pour le traitement des répétitions ; les horodatages sont utilisés à la place. Cela exige que les horloges soient synchronisées. La synchronisation requise dépend du nombre de messages qui peuvent être mis en antémémoire (noter cependant que l'antémémoire de répétition ne contient que des messages qui ont été bien authentifiés). Si on pouvait supposer une antémémoire illimitée, les terminaux n'auraient pas du tout besoin d'être synchronisés (car l'antémémoire pourrait alors contenir tous les messages antérieurs). Cependant, si il y a des restrictions sur la taille de l'antémémoire de répétition, les horloges devront être synchronisées d'une certaine façon. En bref, on peut dire en général qu'il y a un compromis entre la taille de l'antémémoire de répétition et la synchronisation requise.

L'usage de l'horodatage empêche les attaques en répétition sous les hypothèses suivantes :

* Chaque hôte a une horloge qui est au moins "en synchronisation lâche" avec les horloges des autres hôtes.
* Si les horloges sont synchronisées par le réseau, un protocole sûr de synchronisation d'horloge du réseau DEVRAIT être utilisé, par exemple, [ISO3].
* Chaque répondeur utilise une antémémoire de répétition afin de se souvenir des messages bien authentifiés présentés dans un biais d'horloge admissible (qui est réglé par la politique locale).
* Les messages répétés et périmés, par exemple, les messages qu'on peut trouver dans l'antémémoire de répétition ou qui ont un horodatage périmé sont éliminés et ne sont pas traités.
* Si l'hôte perd la trace des demandes entrantes (par exemple, à cause d'une surcharge) il rejette toutes les demandes entrantes jusqu'à ce que soit passé l'intervalle de biais d'horloge.

Dans un scénario client-serveur, les serveurs peuvent rencontrer une forte charge de travail, en particulier si une antémémoire de répétition est nécessaire. Cependant, les serveurs qui assument le rôle d'initiateurs MIKEY ne vont pas avoir besoin de gérer d'antémémoire de répétition significative car ils vont refuser tous les messages entrants qui ne sont

pas une réponse à un message précédemment envoyé par le serveur.

En général, un client ne va pas s'attendre à une très grosse charge de messages entrants et peut donc permettre un degré de souplesse de l'ordre de plusieurs minutes ou heures. Si une attaque de déni de service est lancée et si l'antémémoire de répétition devient trop grosse, MIKEY PEUT diminuer de façon dynamique cette souplesse afin que l'antémémoire de répétition reste gérable. Cependant, noter que de telles attaques de déni de service ne seeaient effectuées que par des homologues qui peuvent s'authentifier. Donc, une telle attaque est très facile à retracer et à contrer.

Le nombre maximum de messages qu'un client va avoir besoin de mettre en antémémoire peut varier selon la capacité du client lui-même et du réseau. Le nombre de messages attendus devrait être pris en compte.

Par exemple, supposons qu'on puisse consacrer au plus 6 kO à une antémémoire de répétition. Supposons de plus qu'on ait besoin de mémoriser 30 octets pour chaque message entrant authentifié (le hachage du message fait 20 octets). Cela implique qu'il est possible de mettre en antémémoire approximativement 204 messages. Si le nombre attendu de messages par minute peut être estimé, le biais d'horloge peut facilement être calculé. Par exemple, dans un scénario SIP où le client est supposé, dans le cas le plus extrême, recevoir 10 appels par minute, le biais d'horloge nécessaire est alors approximativement de 20 minutes. Dans un réglage moins extrême, où on peut attendre un appel entrant toutes les cinq minutes, il en résulterait un biais d'horloge de l'ordre de 16,5 heures (approximativement 1000 minutes).

Considérons un cas très extrême, où le nombre maximum de messages entrant est supposé être de l'ordre de 120 messages par minute, et une exigence que le biais d'horloge soit de l'ordre de 10 minutes ; une antémémoire de 48 koctets serait nécessaire.

Donc, on peut noter que le biais d'horloge requis va dépendre largement des réglages utilisés avec MIKEY. Une recommandation est de fixer une taille à l'antémémoire de répétition, permettant un grand biais d'horloge (le biais d'horloge initial peut être réglé en fonction de l'application dans laquelle elle est utilisée). Avec l'augmentation de l'antémémoire de répétition, le biais d'horloge est diminué selon le pourcentage utilisé de l'antémémoire de répétition. Noter que ceci est traité en local, ce qui n'exigera pas d'interaction avec l'homologue (même si cela peut l'affecter indirectement). Cependant, la façon exacte dont est mise en œuvre une telle fonctionnalité sort du domaine d'application du présent document et est considéré comme spécifique de la mise en œuvre.

En cas d'une attaque de DoS, le client va très vraisemblablement être capable de gérer l'antémémoire de répétition. Une attaque de DoS plus probable (et plus sérieuse) est une attaque du CPU où l'agresseur envoie des messages à l'homologue, qui doit alors consommer des ressources pour vérifier les MAC/signatures des messages entrants.

6. Codage de charge utile

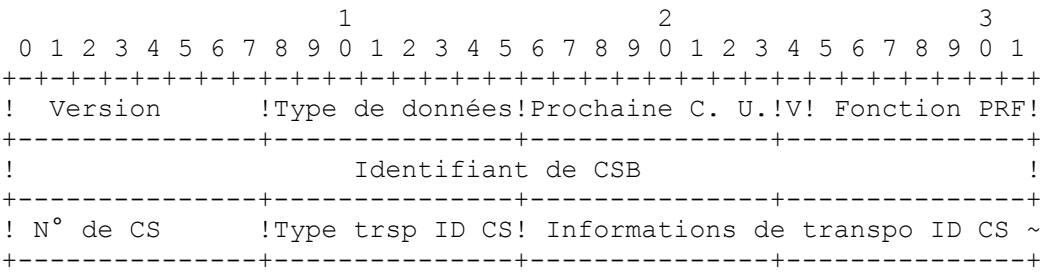
La présente section décrit en détails toutes les charges utiles. Pour tous les codages, l'ordre des octets du réseau est toujours utilisé. Elle définit les types pris en charge (par exemple, les fonctions de hachage) les types dont la mise en œuvre est obligatoire, ainsi que les types par défaut (noter que la mise en œuvre des types par défaut est aussi obligatoire). La prise en charge de tous les autres types est implicitement supposée facultative.

Dans la suite de ce document, on notera qu'est définie la prise en charge de SRTP [RFC3711] comme protocole de sécurité. Cela va nous aider à mieux comprendre l'objet des différentes charges utiles et champs. D'autres protocoles de sécurité PEUVENT être spécifiés pour utilisation dans MIKEY, voir la Section 10.

Dans ce qui suit, le signe ~ indique des champs de longueur variable.

6.1 Charge utile En-tête commun (HDR)

La charge utile En-tête commun DOIT toujours être présente comme première charge utile dans chaque message. L'en-tête commun comporte une description générale du message d'échange.



* Version (8 bits) : c'est le numéro de version de MIKEY. version = 0x01 se réfère à MIKEY défini ici.

* Type de données (8 bits) : décrit le type de message (par exemple, message de transport de clé publique, message de vérification, message d'erreur).

Type de données	Valeur	Commentaire
Prépartagé	0	Message de clé prépartagée de l'initiateur
Mess vérif PSK	1	Message de vérification d'un message de clé prépartagée
Clé publique	2	Message de transport de clé publique de l'initiateur
Mess vérif PK	3	Message de vérification d'un message de clé publique
Init D-H	4	Message d'échange DH de l'initiateur
Rép D-H	5	Message d'échange DH du répondeur
Erreur	6	Message d'erreur

Tableau 6.1.a

* Prochaine C. U. (8 bits) : identifie la charge utile qui est ajoutée après cette charge utile.

Prochaine charge utile	Valeur	paragraphe
Dernière charge utile	0	-
KEMAC	1	6.2
PKE	2	6.3
DH	3	6.4
SIGN	4	6.5
T	5	6.6
ID	6	6.7
CERT	7	6.7
CHASH	8	6.8
V	9	6.9
SP	10	6.10
RAND	11	6.11
ERR	12	6.12
Données de clé	20	6.13
General Ext.	21	6.15

Tableau 6.1.b

Noter que certaines charges utiles ne peuvent pas suivre directement l'en-tête (comme "Dernière charge utile", "Signature"). Cependant, le champ Prochaine charge utile est générique pour toutes les charges utiles. Donc, une valeur est allouée pour chaque charge utile. Le champ Prochaine charge est réglé à zéro (Dernière charge utile) si la charge utile en cours est la dernière.

* V (1 bit) : fanion pour indiquer si un message de vérification est ou non attendu (cela n'a de signification que lorsque il est établi par l'initiateur). Le fanion V DEVRA être ignoré par le receveur dans la méthode DH (car la réponse est OBLIGATOIRE).

  V = 0 ==> pas de réponse attendue
  V = 1 ==> réponse attendue

* Fonction PRF (7 bits) : indique la fonction PRF qui a été ou sera utilisée pour la déduction de clé.

Fonction PRF	Valeur	Commentaires
MIKEY-1	0	Obligatoire (voir le paragraphe 4.1.2)

Tableau 6.1.c

* CSB ID (32 bits) : identifie le CSB. Il est RECOMMANDÉ que l'ID de CSB soit choisi au hasard par l'initiateur. Cet ID DOIT être univoque pour chaque paire d'initiateur-répondeur, c'est-à-dire, pas nécessairement unique au monde. Un initiateur DOIT chercher les collisions lors du choix de l'ID (si l'initiateur a déjà un ou plusieurs CSG établis avec le répondeur). Le répondeur utilise le même ID de CSB dans la réponse.

* N° de CS (8 bits) : indique le nombre de sessions de chiffrement qui seront traitées au sein du CBS. Note que bien qu'il soit possible d'utiliser 255 CS, il est peu probable qu'un CSB en comporte autant. L'entier 0 est interprété comme pas de CS incluse. Cela peut être le cas dans un message d'établissement initial.

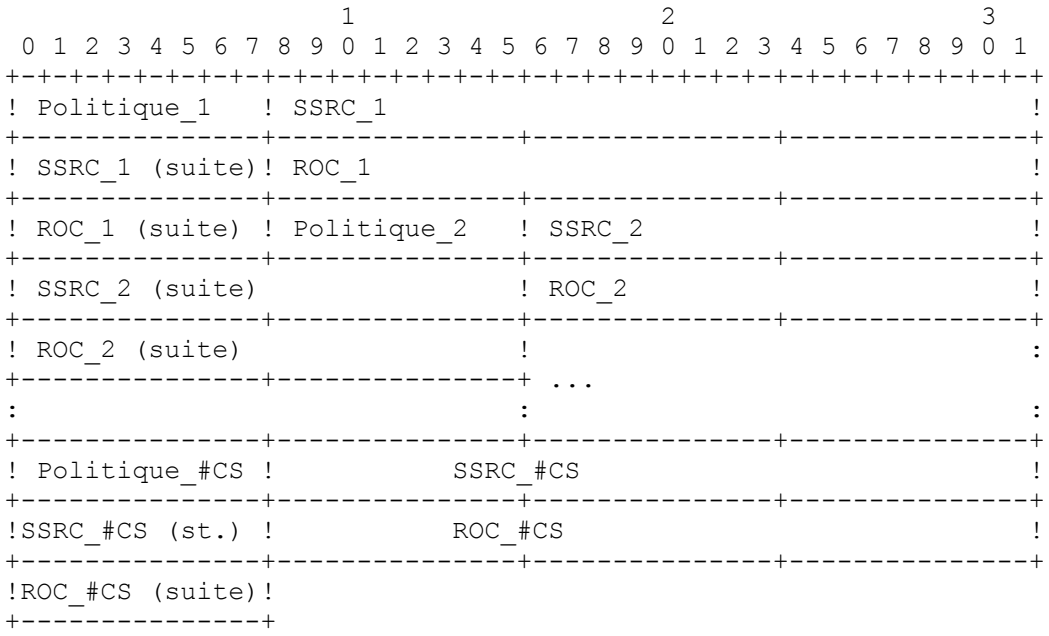
* Type trsp ID CS (8 bits) : spécifie la méthode de transposition univoque des sessions de chiffrement en sessions de protocole de sécurité.

Type trsp ID CS	**Valeur**
SRTP-ID	0

Tableau 6.1.d

* Informations de transpo ID CS (longueur variable) : identifie la ou les sessions de chiffrement pour lesquelles la SA devrait être créée. Le type de transposition actuellement défini est le SRTP-ID (défini au paragraphe 6.1.1).

6.1.1 SRTP ID



* Politique_i (8 bits) : c'est la politique de sécurité appliquée au flux avec SSRC_i. La même politique de sécurité peut s'appliquer pour toutes les CS.

* SSRC_i (32 bits) : spécifie le SSRC qui DOIT être utilisé pour le i ème flux SRTP. Noter que c'est l'envoyeur des flux qui choisit le SSRC. Donc, il est possible que l'initiateur de MIKEY ne puisse pas remplir tous les champs. Dans ce cas, les SSRC qui ne sont pas choisis par l'initiateur sont réglés à zéro et le répondeur remplit ces champs dans le message de réponse. Noter que les exigences spécifiques de SRTP sur l'unicité des SSRC (pour éviter les problèmes de deux bourrages si la même TEK est utilisée pour plus d'un flux) [RFC3711].

* ROC_i (32 bits) : compteur de roulement actuel utilisé dans SRTP. Si la session SRTP n'a pas commencé, ce champ est à 0. Ce champ est utilisé pour permettre à un membre de se joindre et se synchroniser à un flux déjà commencé.

Note : Le flux qui utilise SSRC_i va aussi avoir l'identifiant de session de chiffrement égal au nombre i (NON au SSRC).

6.2 Charge utile Transport de données de clé (KEMAC)

La charge utile Transport de données de clé contient des sous charges utiles de données de clé chiffrées (voir au paragraphe 6.13 la définition de la sous charge utile de données de clé). Elle peut contenir une ou plusieurs charges utiles de données de clé, chacune comportant, par exemple, une TGK. La dernière charge utile de données de clé a son champ

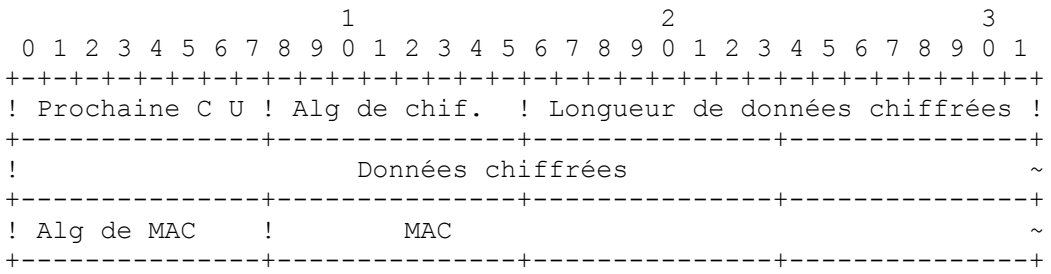
Prochaine charge utile réglé à Dernière charge utile. Pour un message de mise à jour (voir aussi le paragraphe 4.5) il est permis de sauter les sous charges utiles de données de clé (d'où il va résulter que la longueur des données chiffrées sera égale à 0).

Noter que la couverture du MAC dépend de la méthode utilisée, c'est-à-dire, clé prépartagée ou publique, voir ci-dessous.

Si la méthode de transport utilisée est celle des clés prépartagées, cette charge utile Transport de données de clé est la dernière charge utile dans le message (noter que le champ Prochaine charge utile est réglé à Dernière charge utile). Le MAC est alors calculé sur le message MIKEY entier suivant les directives du paragraphe 5.2.

Si la méthode de transport utilisée est celle de la clé publique, l'identité de l'initiateur est ajoutée dans les données chiffrées. Cela se fait en ajoutant l'identifiant de charge utile comme première charge utile, qui est alors suivie par les sous charges utiles Données de clé. Noter que pour un message de mise à jour, l'identifiant est encore envoyé chiffré au répondeur (cela pour éviter certaines attaques de redirection) même si aucune sous charge utile Données de clé n'est ajoutée derrière.

Dans le cas de clé publique, la couverture du champ MAC est seulement sur la charge utile Transport de données de clé, au lieu du message MIKEY complet comme dans le cas prépartagé. Le MAC est donc calculé sur la charge utile Transport de données de clé, sauf pour le champ MAC où le champ Prochaine charge utile a été réglé à zéro (voir aussi le paragraphe 5.2).



* Prochaine charge utile (8 bits) : identifie la charge utile ajoutée après cette charge utile. Voir les valeurs définies en 6.1.

* Alg de chif. (8 bits) : algorithme de chiffrement utilisé pour chiffrer le champ Données chiffrées.

Algorithme de chiffrement	Valeur	Commentaire
NUL	0	Usage très restreint, voir le paragraphe 4.2.3 !
AES-CM-128	1	Obligatoire ; AES-CM avec une clé de 128 bits, (voir en 4.2.3)
AES-KW-128	2	Clé enveloppe AES avec une clé de 128 bits, (voir en 4.2.3)

Tableau 6.2.a

* Longueur de données chiffrées (16 bits) : Longueur des données chiffrées (en octets).

* Données chiffrées (longueur variable) : sous-charges utiles de clé chiffrée (voir au paragraphe 6.13).

* Algorithme de MAC (8 bits) : spécifie l'algorithme d'authentification utilisé.

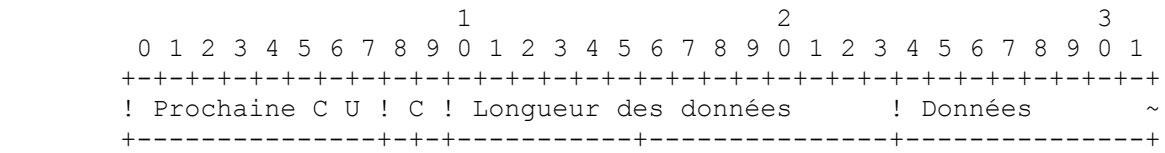
Algorithme de MAC	Valeur	Commentaires	Longueur (bits)
NUL	0	Usage limité, paragraphe 4.2.4	0
HMAC-SHA-1-160	1	Obligatoire, paragraphe 4.2.4	160

Tableau 6.2.b

* MAC (Longueur variable) : code d'authentification de message sur le message entier.

6.3 Charge utile Données d'enveloppe (PKE)

La charge utile Données d'enveloppe contient la clé d'enveloppe chiffrée qui est utilisée dans le transport de clé publique pour protéger les données dans la charge utile Transport de données de clé. L'algorithme de chiffrement utilisé est impliqué par le certificat/clé publique utilisé.



* Prochaine charge utile (8 bits) : identifie la charge utile ajoutée après cette charge utile. Voir les valeurs en 6.1.

* C (2 bits) ; indicateur d'antémémoire de clé d'enveloppe (paragraphe 3.2).

Type d'antémémoire	Valeur	Commentaires
pas d'antémémoire	0	La clé d'enveloppe NE DOIT PAS être mise en antémémoire
antémémoire	1	La clé d'enveloppe DOIT être mise en antémémoire
antémémoire pour CSB	2	La clé d'enveloppe DOIT être mise en antémémoire, mais seulement pour être utilisée pour le CSB spécifique.

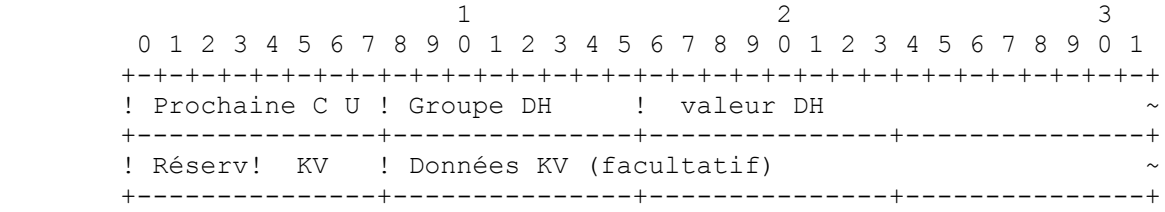
Tableau 6.3

* Longueur des données (14 bits) : Longueur du champ Données (en octets).

* Données (Longueur variable) : Clé d'enveloppe chiffrée.

6.4 Charge utile Données DH (DH)

La charge utile Données DH porte la valeur DH et indique le groupe DH utilisé. On notera que dans ce paragraphe, "OBLIGATOIRE" est conditionné à la prise en charge de DH.



* Prochaine C U (8 bits) : identifie la charge utile ajoutée après cette charge utile. Voir les valeurs au paragraphe 6.1.

* Groupe DH (8 bits) : identifie le groupe DH utilisé.

Groupe DH	Valeur	Commentaire	Longueur (en bits) de valeur DH
OAKLEY 5	0	Obligatoire	1536
OAKLEY 1	1		768
OAKLEY 2	2		1024

Tableau 6.4

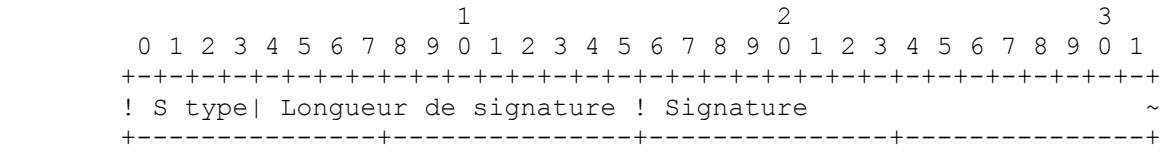
* valeur DH (longueur variable) : valeur DH publique (la longueur est impliquée par le groupe utilisé).

* KV (4 bits) : indique le type de période de validité de clé spécifiée. Cela peut être fait en utilisant un SPI (ou un MKI dans SRTP) ou en fournissant un intervalle dans lequel la clé est valide (par exemple, dans ce dernier cas, pour SRTP ce sera la gamme d'indice dans laquelle la clé est valide). Voir les valeurs prédéfinies au paragraphe 6.13.

* Données KV (longueur variable) : cela inclut le SPI/MKI ou un intervalle (voir au paragraphe 6.14). Si KV est NUL, ce champ n'est pas inclus.

6.5 Charge utile Signature (SIGN)

La charge utile Signature porte la signature et les données qui s'y rapportent. La charge utile signature est toujours la dernière charge utile dans les messages de transport PK et d'échange DH. L'algorithme de signature utilisé est implicite d'après le certificat/clé publique utilisé.



* S type (4 bits) : indique l'algorithme de signature appliqué par le signataire.

S type	Valeur	Commentaires
RSA/PKCS#1/1.5	0	Obligatoire, signature PKCS n° 1 version 1.5 [PSS]
RSA/PSS	1	Signature RSASSA-PSS [PSS]

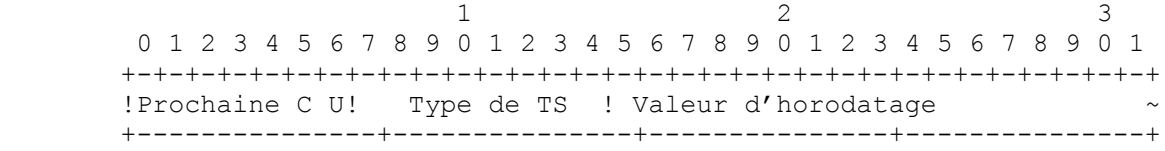
Tableau 6.5

* Longueur de signature (12 bits) : c'est la longueur du champ Signature (en octets).

* Signature (longueur variable) : c'est la signature (son format et le bourrage dépendent du type de signature).

6.6 Charge utile Horodatage *(TS, Timestamp)*

La charge utile Horodatage porte les informations d'horodatage.



* Prochaine C U (8 bits) : identifies la charge utile qui est ajoutée après cette charge utile. Voir les valeurs en 6.1.

* Type de TS (8 bits) : spécifie le type d'horodatage utilisé.

Type de TS	Valeur	Commentaires	Longueur la valeur de l'horodatage
NTP-UTC	0	Obligatoire	64 bits
NTP	1	Obligatoire	64 bits
COMPTEUR	2	Facultatif	32 bits

Tableau 6.6

Note : COMPTEUR DEVRA être bourré (avec des zéros en tête) jusqu'à une valeur de 64 bits lorsque utilisé comme entrée pour la PRF par défaut.

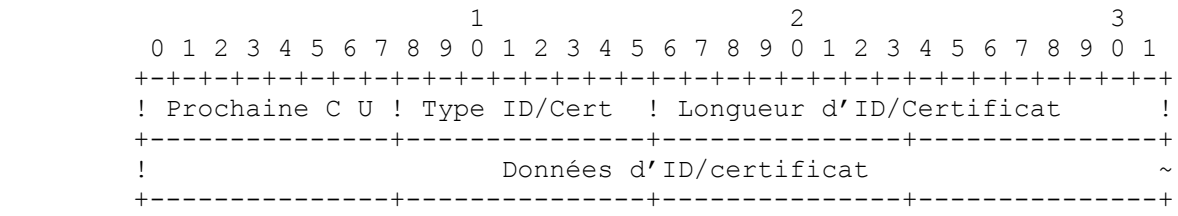
* Valeur d'horodatage (longueur variable) : C'est la valeur de l'horodatage du type spécifié.

6.7 Charge utile ID (ID) / charge utile Certificat (CERT)

Noter que la charge utile ID et la charge utile Certificat sont deux charges utiles complètement différentes (qui ont des identifiants de charge utile différents). Cependant, comme elles partagent la même structure, elles sont décrites dans le même paragraphe.

La charge utile ID porte un identifiant à définition univoque.

La charge utile certificat contient un indicateur du certificat fourni ainsi que les données de certificat. Si une chaîne de certificats doit être fournie, chaque certificat dans la chaîne devrait être inclus dans une charge utile CERT distincte.



* Prochaine charge utile (8 bits) : identifie la charge utile qui est ajoutée après cette charge utile. Voir les valeurs en 6.1.

Si la charge utile est un ID de charge utile, les valeurs suivantes s'appliquent au champ Type d'ID :

* Type ID (8 bits) : spécifie le type d'identifiant utilisé.

ID Type	Valeur	Commentaires
NAI	0	Obligatoire (voir la [RFC2486])
URI	1	Obligatoire (voir la [RFC2396])

Tableau 6.7.a

Si la charge utile est un certificat, les valeurs suivantes s'appliquent pour le champ Type Cert:

* Type Cert (8 bits) : spécifie le type de certificat utilisé.

Type Cert	Valeur	Commentaires
X.509v3	0	Obligatoire
X.509v3 URL	1	URL en ASCII de la localisation du
CertificatX.509v3 Sign	2	Obligatoire (utilisé seulement pour les signatures)
X.509v3 Encr	3	Obligatoire (utilisé seulement pour le chiffrement)

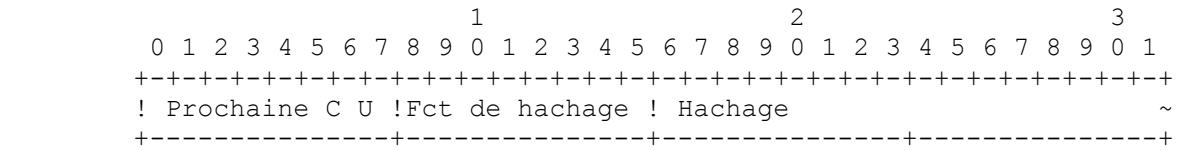
Tableau 6.7.b

* Longueur d'ID/Cert (16 bits) : c'est la longueur du champ ID ou Certificat (en octets).

* Données d'ID/Certificat (longueur variable) : ce sont les données qui constituent l'identifiant ou le certificat. Les certificats X.509 [RFC3280] sont inclus comme des chaînes d'octets qui utilisent le codage DER comme spécifié dans X.509.

6.8 Charge utile Hachage Cert (CHASH)

La charge utile Hachage Cert contient le hachage du certificat utilisé.



* Prochaine charge utile (8 bits) : identifie la charge utile qui est ajoute après cette charge utile. Voir les valeurs en 6.1.

* Fct de hachage (8 bits) : indique la fonction de hachage qui est utilisée (voir aussi le paragraphe 4.2.1).

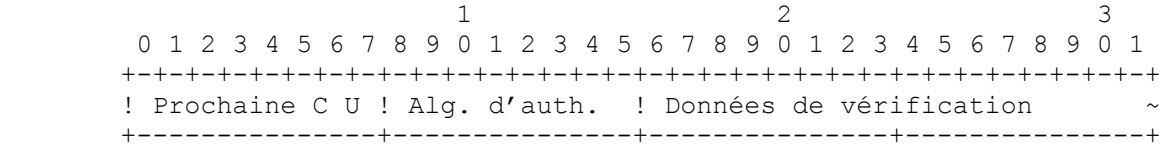
Fonction de hachage	Valeur	Commentaire	Longueur du hachage (bits)
SHA-1	0	Obligatoire	160
MD5	1	|	128

Tableau 6.8

* Hachage (longueur variable) : ce sont les données du hachage. La longueur est impliquée par la fonction utilisée.

6.9 Charge utile Msg Ver (V)

La charge utile Msg Ver contient le message de vérification calculé dans la clé prépartagée et les méthodes de transport de clé publique. Noter que le MAC est calculé sur le message MIKEY entier, ainsi que les ID et l'horodatage (voir aussi le paragraphe 5.2).



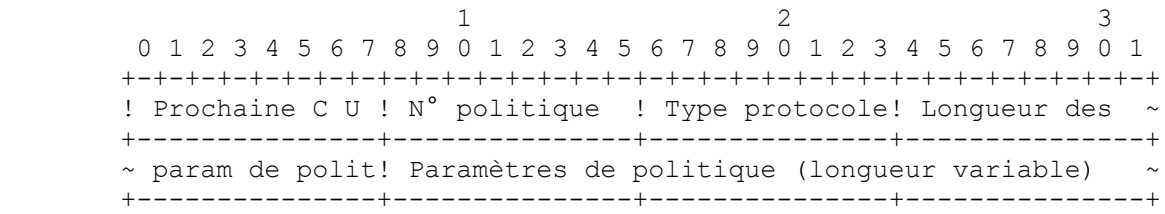
* Prochaine C U (8 bits) : identifie la charge utile qui est ajoutée après cette charge utile. Voir les valeurs en 6.1.

* Alg. d'auth. (8 bits) : spécifie l'algorithme de MAC utilisé pour le message de vérification. Voir les valeurs définies au paragraphe 6.2.

* Données de vérification (longueur variable) : données du message de vérification. La longueur est impliquée par l'algorithme d'authentification utilisé.

6.10 Charge utile Politique de sécurité (SP)

La charge utile Politique de sécurité définit un ensemble de politiques qui s'applique à un protocole de sécurité spécifique.



* Prochaine charge utile (8 bits) : identifie la charge utile ajoutée après cette charge utile. Voir les valeurs en 6.1.

* N° de politique (8 bits) : chaque charge utile Politique de sécurité doit recevoir de l'homologue local un numéro distinct pour la session MIKEY en cours. Ce numéro est utilisé pour transposer une session de chiffrement en une politique spécifique (voir aussi au paragraphe 6.1.1).

* Type protocole (8 bits) : définit le protocole de sécurité.

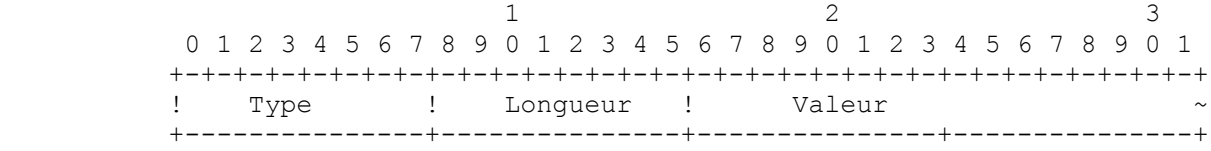
Type de protocole	Valeur
SRTP	0

Tableau 6.10

* Longueur des paramètres de polit (16 bits) : définit la longueur totale des paramètres de politique pour le protocole de sécurité spécifié.

* Paramètres de politique (longueur variable) : définit la politique pour le protocole de sécurité spécifié.

La partie paramètres de politique est construite sur un ensemble de champs Type/Longueur/Valeur. Pour chaque protocole de sécurité est défini un ensemble de types/valeurs possibles qui peuvent être négociés.



* Type (8 bits) : spécifie le type du paramètre.
* Longueur (8 bits) : spécifie la longueur du champ Valeur (en octets).
* Valeur (longueur variable) : spécifie la valeur du paramètre.

6.10.1 Politique SRTP

Cette politique spécifie les paramètres pour SRTP et SRTCP. Les types/valeurs qui peuvent être négociés sont définis par le tableau suivant :

Type	Signification	Valeurs possibles
0	algorithme de chiffrement	voir ci-dessous
1	longueur de clé de chiffrement de session	dépend du chiffrement utilisé
2	algorithme d'authentification	voir ci-dessous
3	longueur de clé d'authentification de session	dépend du MAC utilisé
4	longueur de clé de salage de session	voir les recommandations dans la [RFC3711]
5	fonction SRTP pseudo aléatoire	voir ci-dessous
6	taux de déduction de clé	voir les recommandations dans la [RFC3711]
7	chiffrement SRTP marche/arrêt	0 si arrêt, 1 si marche
8	chiffrement SRTCP marche/arrêt	0 si arrêt, 1 si marche
9	ordre de FEC de l'envoyeur	voir ci-dessous
10	authentification SRTP marche/arrêt	0 si arrêt, 1 si marche
11	longueur d'étiquette d'authentification	en octets
12	longueur du préfixe SRTP	en octets

Tableau 6.10.1.a

Noter que si un Type/Valeur n'est pas établi, la valeur par défaut est utilisée (selon les propres critères de SRTP). Noter aussi que, si "Longueur de clé de chiffrement de session" est établi, cela devrait aussi être vu comme la longueur de la clé maîtresse (autrement, on utilise la longueur de clé maîtresse par défaut de SRTP).

Pour l'algorithme de chiffrement, une longueur de un octet est suffisante. Les valeurs possibles actuellement définies sont :

Algorithme de chiffrement SRTP	Valeur
NUL	0
AES-CM	1
AES-F8	2

Tableau 6.10.1.b

où AES-CM est AES en mode CM, et AES-F8 est AES en mode f8 [RFC3711].

Pour l'algorithme d'authentification, une longueur d'un octet est suffisante. Les valeurs possibles actuellement définies sont :

Algorithme d'authentification SRTP	Valeur
NUL	0
HMAC-SHA-1	1

Tableau 6.10.1.c

Pour la fonction pseudo aléatoire de SRTP, une longueur d'un octet est suffisante. Les valeurs possibles actuellement définies sont :

PRF SRTP	Valeur
AES-CM	0

Tableau 6.10.1.d

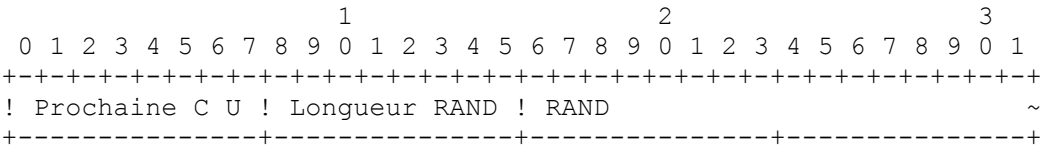
Si la correction d'erreur directe (FEC) est utilisée en même temps que SRTP, MIKEY peut négocier l'ordre dans lequel ces facilités devraient être appliquées du côté de l'envoyeur.

Ordre de FEC	Valeur	Commentaires
FEC-SRTP	0	FEC d'abord, puis SRTP

Tableau 6.10.1.e

6.11 Charge utile RAND (RAND)

La charge utile RAND consiste en une chaîne binaire (pseudo-)aléatoire. RAND DOIT être généré par CSB de façon indépendante (noter que si le CSB a plusieurs membres, l'initiateur DOIT utiliser le même RAND pour tous les membres). Pour les recommandations d'aléa pour la sécurité, voir la [RFC1750].



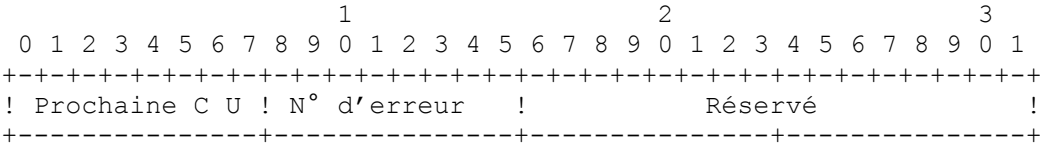
* Prochaine charge utile (8 bits) : identifie la charge utile qui est ajoutée après cette charge utile. Voir les valeurs en 6.1.

* Longueur RAND (8 bits) : longueur de RAND (en octets). Elle DEVRAIT être d'au moins 16.

* RAND (longueur variable) : chaîne binaire (pseudo-)aléatoire.

6.12 Charge utile Erreur (ERR)

La charge utile Erreur est utilisée pour spécifier les erreurs qui peuvent survenir.



* Prochaine charge utile (8 bits) : identifie la charge utile qui est ajoutée après cette charge utile. Voir les valeurs en 6.1.

* N° d'erreur (8 bits) : indique le type de l'erreur qui a été rencontrée.

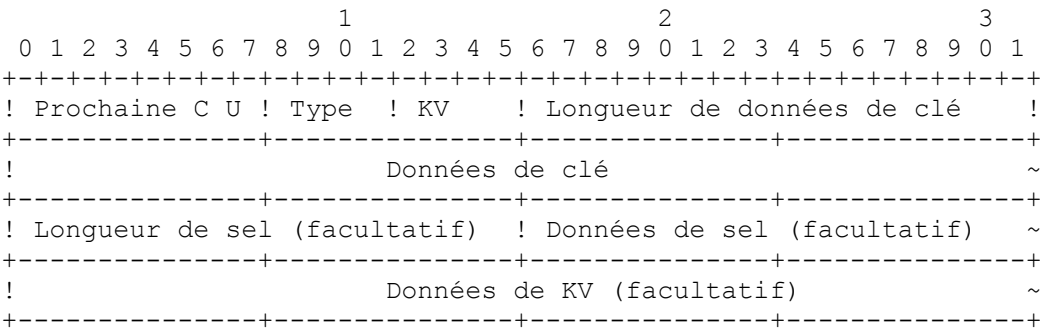
Nom d'erreur	Valeur	Commentaire
Échec auth	0	Échec d'authentification
TS invalide	1	Horodatage invalide
PRF invalide	2	Fonction PRF non acceptée
MAC invalide	3	Algorithme de MAC non accepté
EA invalide	4	Algorithme de chiffrement non accepté
HA invalide	5	Fonction de hachage non acceptée
DH invalide	6	Groupe DH non accepté
ID invalide	7	Identifiant non accepté
Cert invalide	8	Certificat non accepté
SP invalide	9	Type SP non accepté
Sppar invalide	10	Paramètres SP non acceptés
DT invalide	11	Type de données non accepté
Non spécifié	12	Une erreur non spécifié est survenue

Tableau 6.12

6.13 Sous charge utile Données de clé

La charge utile Données de clé contient du matériel de clés, par exemple, des TGK. Les charges utiles Données de clé ne sont jamais incluses en clair, mais comme partie chiffrée de la charge utile Transport de données de clé.

Noter qu'une charge utile Transport de données de clé peut contenir plusieurs sous charges utiles Données de clé.



* Prochaine charge utile (8 bits) : identifie la charge utile ajoutée après cette charge utile. Voir les valeurs en 6.1.

* Type (4 bits) : indique le type de clé incluse dans la charge utile.

Type	**Valeur**
TGK	0
TGK+SEL	1
TEK	2
TEK+SEL	3

Tableau 6.13.a

Noter qu'est fournie la possibilité d'inclure une TEK (au lieu d'utiliser la TGK). Lorsque elle est envoyée directement, la TEK ne peut généralement pas être partagée entre plus d'une session de chiffrement (sauf si le protocole de sécurité le permet, par exemple, [RFC3711]). L'usage recommandé d'envoyer une TEK, au lieu d'une TGK, est lorsque il existe du matériel pré chiffré et donc que la TEK est connue à l'avance.

* KV (4 bits) : indique le type de période de validité de la clé spécifiée. Cela peut être fait en utilisant un SPI (ou MKI dans le cas de la [RFC3711]) ou en fournissant un intervalle dans lequel la clé est valide (par exemple, dans le premier cas, pour SRTP ce sera la gamme d'indices où la clé est valide).

KV	**Valeur**	**Commentaires**
Nul	0	Pas de règle d'usage spécifique (par exemple, une TEK qui n'a pas de durée de vie spécifique)
SPI	1	La clé est associée au SPI/MKI
Intervalle	2	La clé a une heure de début et d'expiration (par exemple, une TEK SRTP)

Tableau 6.13.b

Noter que lorsque NUL est spécifié, tout SPI ou intervalle est valide. Pour un intervalle, cela signifie que la clé est valide depuis le premier numéro de séquence observé jusqu'à ce que la clé soit remplacée (ou que le protocole de sécurité soit clos).

* Longueur de données de clé (16 bits) : Longueur du champ Données de clé (en octets). Noter que la somme de la longueur globale de toutes les charges utiles Données de clé contenues dans une seule charge utile Transport de données de clé (KEMAC) DOIT être telle que la charge utile KEMAC n'excède pas une longueur de 2^16 octets (longueur totale de KEMAC, voir le paragraphe 6.2).

* Données de clé (longueur variable) : Données de la TGK ou de la TEK.

* Longueur de sel (16 bits) : longueur de la clé de salage en octets. Noter que ce champ n'est inclus que si le sel est spécifié dans le champ Type.

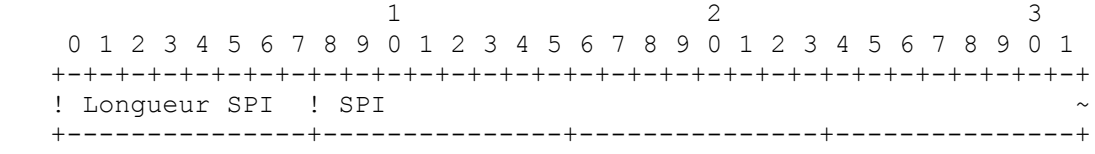
* Données de sel (longueur variable) : Données de la clé de salage. Noter que ce champ n'est inclus que si le sel est spécifié dans le champ Type. (Pour SRTP, c'est ce qu'on appelle le maître sel.)

* Données de KV (longueur variable) : cela inclut soit le SPI, soit un intervalle (voir au paragraphe 6.14). Si KV est NUL, ce champ n'est pas inclus.

6.14 Données de validité de clé

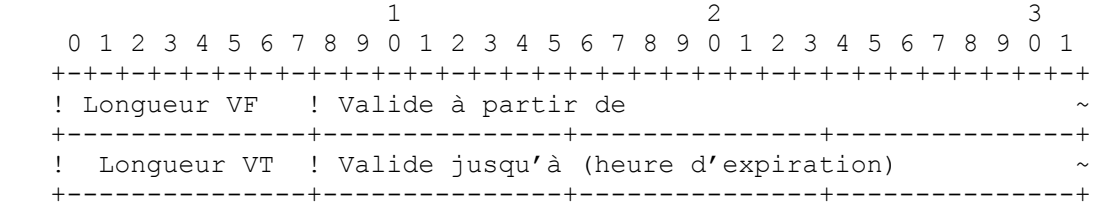
Les données de validité de clé ne sont pas une charge utile autonome, mais font partie soit de la charge utile Données de clé (voir le paragraphe 6.13) soit de la charge utile DH (voir le paragraphe 6.4). Les données de validité de clé donnent une ligne directrice sur le moment où la clé devrait être utilisée. Deux types de KV sont définis (voir au paragraphe 6.13) SPI/MKI (SPI) ou une gamme de durées de vie (intervalle).

SPI/MKI



* Longueur SPI (8 bits) : Longueur du SPI (ou MKI) en octets.
* SPI (longueur variable) : la valeur du SPI (ou MKI).

Intervalle

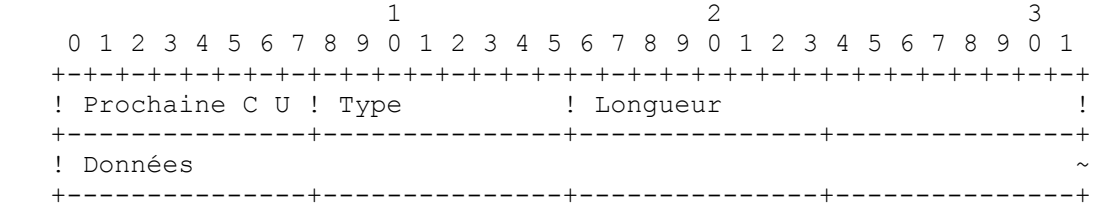


* Longueur VF (8 bits) : longueur du champ Valide à partir de, en octets.
* Valide à partir de (longueur variable) : numéro de séquence, indice, horodatage, ou autre valeur de départ que le protocole de sécurité utilise pour identifier la position de départ de l'usage de la clé.
* Longueur VT (8 bits) : longueur du champ Valide jusqu'à en octets.
* Valide jusqu'à (longueur variable) : numéro de séquence, indice, horodatage, ou autre valeur de terminaison que le protocole de sécurité peut utiliser pour identifier l'expiration de l'usage de la clé.

Noter que pour l'usage de SRTP, la période de validité de la clé pour une TGK/TEK devrait être spécifiée avec un intervalle, où la longueur VF/VT est égale à 6 octets (c'est-à-dire, la taille de l'indice) ou avec un MKI. Il est RECOMMANDÉ que si plus d'un flux SRTP partage les mêmes clés et si la mise à jour ou le changement des clés est désiré, ceci soit traité avec MKI plutôt que par la méthode From-To *(de-à)*.

6.15 Charge utile Extension générale

La charge utile Extensions générale est incluse pour permettre de possibles extensions à MIKEY sans qu'il soit besoin de définir à chaque fois un nouveau protocole complet. Cette charge utile peut être utilisée dans tout message MIKEY et fait partie de la partie Données authentifiées/signées.



* Prochaine charge utile (8 bits) : identifie la charge utile ajoutée après cette charge utile.
* Type (8 bits): identifie le type de charge utile générale.

Type	Valeur	Commentaires
ID de fabricant	0	Chaîne d'octets spécifique du fabricant
Identifiants SDP	1	Liste des identifiants de gestion de clé SDP (alloués dans la [RFC4567])

Tableau 6.15

* Longueur (16 bits) : Longueur en octets du champ Données.
* Données (Longueur variable) : Données générales de charge utile.

7. Protocoles de transport

MIKEY PEUT être intégré dans des protocoles d'établissement de session. Actuellement, l'intégration de MIKEY au sein de SIP/SDP et RTSP est définie dans la [RFC4567]. MIKEY PEUT utiliser d'autres transports, mais il faut définir comment MIKEY est transporté sur un tel protocole de transport.

8. Groupes

Ce qu'on a exposé jusqu'à maintenant ne se limite pas à une seule communication d'homologue à homologue (sauf pour la méthode DH) mais peut être utilisé pour distribuer les clés de groupe pour de petits groupes interactifs et des scénarios simples de un à plusieurs. Le paragraphe 2.1 décrit les scénarios dans l'objectif de MIKEY. Cette section décrit comment MIKEY est utilisé dans un scénario de groupe (voir cependant aussi le paragraphe 4.3 sur les questions d'autorisation).

8.1 Simple de un à plusieurs

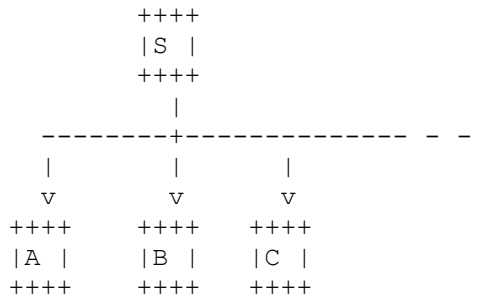


Figure 8.1 : Scénario simple de un à plusieurs

Dans le scénario simple de un à plusieurs, un serveur envoie des flux à un petit groupe de clients. On utilise RTSP ou SIP pour l'enregistrement et pour l'établissement de la gestion de clés. Le serveur de flux agit comme initiateur de MIKEY. Dans ce scénario, le mécanisme de transport de clé prépartagée ou de clé publique sera approprié pour transporter la même TGK à tous les clients (d'où il va résulter des TEK communes pour le groupe).

Note : si les mêmes TGK/TEK devaient être utilisées par tous les membres de groupe, le serveur de flux DOIT spécifier les mêmes CSB_ID et CS_ID pour la session à tous les membres du groupe.

Comme la communication peut être effectuée en utilisant la diffusion groupée, les membres ont besoin d'une politique de sécurité commune si ils veulent faire partie du groupe. Cela limite les possibilités de négociation.

De plus, l'initiateur devrait examiner avec attention s'il faut ou non demander le message de vérification en réponse de chaque receveur, car il peut en résulter une certaine charge pour l'initiateur lui-même lorsque la taille du groupe augmente.

8.2 Groupe interactif de petite taille

Comme décrit dans les généralités, pour les groupes interactifs de petite taille, on peut s'attendre à ce que chaque client soit chargé de l'établissement de la sécurité pour ses flux sortants. Dans ces scénarios, la méthode de transport de clé prépartagée ou de clé publique est utilisée.

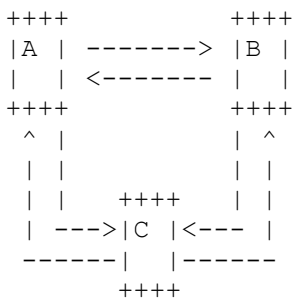


Figure 8.2 : Groupe de petite taille sans contrôleur centralisé

Un scénario peut alors être que le client établisse un appel à trois phases, en utilisant SIP. Du fait de la petite taille du groupe, SRTP en envoi individuel est utilisé entre les clients. Chaque client établit la sécurité pour son ou ses flux sortants avec les autres.

Comme pour le cas de simple un à plusieurs, le client des flux spécifie le ou les mêmes CSB_ID et CS_ID pour ses sessions sortantes si la ou les mêmes TGK/TEK sont utilisée pour tous les membres du groupe.

9. Considérations pour la sécurité

9.1 Généralités

Les protocoles de gestion de clé fondés sur l'horodatage/compteur et un aller-retour de transport de clé ont déjà été normalisés, par exemple à l'ISO [ISO1], [ISO2]. La sécurité générale de ces types de protocoles se trouve dans divers articles et publications , cf. [HAC], [AKE], [LOA].

La force d'une chaîne est celle de son maillon le plus faible. Si on veut un certain niveau de protection, les fonctions cryptographiques qui protègent les clés durant le transport/échange DOIVENT offrir une sécurité correspondant au moins à ce niveau.

Par exemple, si on veut la sécurité contre des attaques d'une complexité de 2^96, on devrait alors choisir un chiffrement symétrique sûr prenant au moins en charge des clés de 96 bits (128 bits peut être un choix pratique) pour la protection réelle du support, et un mécanisme de transport de clé qui fournisse une protection équivalente, par exemple, le transport de clé prépartagée de MIKEY avec une TGK de 128 bits, ou RSA avec des clés de 1024 bits (ce qui selon [LV] correspond au niveau désiré de 96 bits, avec une certaine marge).

En résumé, la taille de clé pour le mécanisme d'échange de clés DOIT être en rapport avec la taille de la TGK échangée afin qu'il offre au moins le niveau requis. Pour des raisons d'efficacité, on DEVRAIT aussi éviter l'outrance dans la sécurité, par exemple, en n'utilisant pas un transport de clé publique avec des clés publiques donnant un niveau de sécurité qui soit supérieur de plusieurs ordres de grandeur à la longueur de la TGK transportée. On se réfère à [LV] pour les recommandations concrètes de taille de clé.

De plus, si les TGK ne sont pas aléatoires (ou pseudo aléatoires) une recherche en force brute peut être facilitée, diminuant encore la taille de clé effective. Donc, on DOIT faire attention à la conception des générateurs (pseudo)aléatoires pour la génération de TGK, voir [FIPS], [RFC1750].

Pour le choix de la fonction de hachage, SHA-1 avec une sortie de 160 bits est celle par défaut. En général, les tailles de hachage devraient être deux fois le "niveau de sécurité", indiquant que SHA-1-256, [SHA256], devrait être utilisé comme fonction par défaut pour le niveau 128 bits. Cependant, du fait des aspects temps réel de ces scénarios dont nous traitons, des tailles de hachage légèrement inférieures à 256 sont acceptable, car la probabilité normale "existentielle" de collision serait d'importance secondaire.

Dans un faisceau de sessions de chiffrement (CSB, *Crypto Session Bundle*) les sessions de chiffrement peuvent partager la même TGK comme on l'a exposé plus haut. Du point de vie de la sécurité, pour satisfaire au critère en cas de partage de la TGK, le chiffrement des sessions cryptographiques individuelles est effectué de façon "indépendante". Dans MIKEY, cela se fait avec des identifiants univoques de session de chiffrement (voir aussi le paragraphe 4.1) et avec une méthode de déduction de TEK qui fournit des TEK cryptographiquement indépendantes aux sessions cryptographiques distinctes (au sein du CSB) sans considération du protocole de sécurité utilisé.

Précisément, les déductions de clés, comme spécifié au paragraphe 4.1, sont mises en œuvre par une fonction pseudo

aléatoire. Celle utilisée ici est une version simplifiée de celle utilisée dans TLS [RFC2246]. Ici, une seule fonction de hachage est utilisée, tandis que TLS utilise deux fonctions différentes. Ce choix est motivé par la haute fiabilité de la fonction de hachage SHA-1, et par l'efficacité et la simplicité de conception (complexité n'implique pas sécurité). Bien sûr, comme le montre [DBJ], si un des deux hachages est sévèrement cassé, la PRF TLS est en fait moins sûre que si un seul hachage avait été utilisé sur la clé entière, comme on le fait dans MIKEY.

Dans les schéma de clé prépartagée et de clé publique, la TGK est générée par une seule partie (l'initiateur). Cela rend MIKEY quelque peu plus sensible si l'initiateur utilise un mauvais générateur de nombres aléatoires. On devrait aussi noter que ni le schéma de clé prépartagée ni celui de la clé publique ne fournissent de parfait secret de transmission. Si on désire la contribution mutuelle ou le secret parfait de transmission, la méthode Diffie-Hellman doit être utilisée. L'authentification (par exemple, des signatures) est exigée dans la méthode Diffie-Hellman pour empêcher les attaques par interposition.

Sécurité avant/arrière : si la TGK est exposée, toutes les TEK générées sont compromises. Cependant, dans l'hypothèse où la fonction de déduction est une fonction pseudo aléatoire, la divulgation d'une TEK individuelle ne compromet pas les autres TEK (antérieures ou ultérieures) déduites de la même TGK. Le mode Diffie-Hellman peut être retenu par les usagers soupçonneux, car il est le seul à prendre en charge ce qu'on appelle le secret de transmission parfait (PFS, *perfect forward secrecy*). À la différence de la compromission d'une clé prépartagée (ou de la clé secrète du mode clé publique) où les sessions futures et les sessions enregistrées depuis le passé sont alors aussi compromises.

L'utilisation de noms occasionnels aléatoires (RAND) dans la déduction de clé est de la plus haute importance pour contrer les attaques hors ligne de pré calcul. Noter cependant que les messages de mise à jour réutilisent le vieux RAND. Cela signifie que l'entropie effective totale de la clé (par rapport aux attaques de pré calcul) pour k mises à jour de clé consécutives, en supposant que les TGK et RAND sont chacune longues de n bits, est d'environ L = n*(k+1)/2 bits, comparé au maximum théorique de n*k bits. En d'autres termes, un effort de travail de 2^L PEUT permettre à l'attaquant d'obtenir toutes les k clés de n bits, ce qui est meilleur qu'une attaque en force brute (sauf quand k = 1). Bien que cela puisse paraître une défaite, on notera d'abord que pour un bon choix de n, la complexité 2^L de l'attaque est de loin hors de portée. De plus, le fait qu'une clé puisse être compromise dans une seule attaque est inhérent au problème de l'échange de clé. Considérons par exemple un usager qui, en utilisant une clé RSA de 1024 bits fixes, échange des clés et communique durant les un ou deux ans de durée de vie de la clé publique. Casser cette seule clé RSA va permettre d'accéder à toutes les clés échangées et par conséquent à toutes les communications de cet usager sur la totalité de la période.

Toutes les transformations prédéfinies dans MIKEY utilisent les algorithmes à la pointe du progrès qui ont subi une large évaluation publique. Une des raisons de l'utilisation de AES-CM à partir de SRTP [RFC3711], est d'avoir la possibilité de limiter le nombre global de modes et algorithmes de chiffrement différents, tout en offrant en même temps un haut niveau de sécurité.

9.2 Durée de vie de clé

Même si la durée de vie d'une TGK (ou TEK) n'est pas spécifiée, on DOIT tenir compte de ce que la transformation de chiffrement dans le protocole de sécurité sous-jacent peut dégénérer d'une certaine façon après une certaine quantité de données chiffrées. Il n'est pas possible de déclarer ici des limites générales de durée de vie de clé universellement applicables ; chaque protocole de sécurité devrait définir de telles quantités maximum et déclancher une procédure de changement de clé avant "l'extinction" de la clé. Par exemple, selon SRTP [RFC3711] la TEK, avec la TGK correspondante, DOIT être changée au moins tous les 2^48 paquets SRTP.

Il reste que ceci n'est qu'une règle approximative. Si le protocole de sécurité utilise un chiffrement "idéal" de bloc de b bits (en mode CBC, en mode compteur ou en mode rétroaction, par exemple, OFB, avec un plein retour de b bits) un comportement dégénéré du flux cryptographique, éventuellement utile pour une attaque, est (avec une probabilité constante) supposé survenir après un total d'environ 2^(b/2) blocs de b bits chiffrés (en utilisant des vecteurs d'initialisation aléatoires). Pour avoir une marge de sécurité, le changement de clé DOIT être déclanché bien avant la limite ci-dessus. Voir les détails dans [BDJR].

Pour l'utilisation d'un chiffrement de flux dédié, on se réfère à l'analyse et la documentation du dit chiffrement dans chaque cas spécifique.

9.3 Horodatages

L'utilisation des horodatages, au lieu du défi-réponse, exige que les systèmes aient des horloges synchronisées. Bien sûr, si deux clients ne sont pas synchrones, ils vont avoir des difficultés à établir la sécurité. La solution actuelle fondée sur l'horodatage a été choisie pour permettre un maximum d'un aller-retour (c'est-à-dire, deux messages) mais assure quand même une protection raisonnable contre la répétition. Une version (sûre) fondées sur le défi-réponse exigerait au moins trois messages. Pour une description détaillée du traitement de l'horodatage et de la répétition dans MIKEY, voir en 5.4.

Les expériences pratiques de Kerberos et autres systèmes fondés sur l'horodatage indiquent qu'il n'est pas toujours nécessaire de synchroniser les terminaux sur le réseau. Une configuration manuelle pourrait être une solution de remplacement faisable dans de nombreux cas (en particulier dans les scénarios où le degré de souplesse est élevé). Cependant, le choix doit être fait en fonction du scénario d'utilisation.

9.4 Protection d'identité

La confidentialité de l'utilisateur est une matière complexe qui peut dans une certaine mesure être assurée par les mécanismes cryptographiques, mais exige aussi la mise en application d'une politique et de diverses autres fonctionnalités. Une facette particulière de la confidentialité est la protection de l'identité de l'usager. Cependant, la protection de l'identité n'était pas un objectif principal de la conception de MIKEY. Une telle caractéristique ajoute plus de complexité au protocole et il n'a donc pas été choisi de l'inclure. Comme il est de toutes façon proposé que MIKEY soit transporté sur, par exemple, SIP, l'identité peut être exposée par cela. Cependant, si le protocole transporteur est sécurisé et fournit aussi la protection de l'identité, MIKEY peut hériter de la même caractéristique. Comment cela devrait être fait fera l'objet d'études à venir.

9.5 Déni de service

Ce protocole est résistant aux attaques de déni de service au sens où un répondeur ne construit aucun état (au niveau du protocole de gestion de clé) avant qu'il ait authentifié l'initiateur. Cependant, ce protocole, comme beaucoup d'autres, est ouvert aux attaques qui utilisent des adresses IP usurpées pour créer un grand nombre de demandes falsifiées. Cela peut, par exemple, être résolu en laissant le protocole qui transporte MIKEY faire un essai de validité d'adresse IP. Le protocole SIP peut fournir cela en utilisant le mécanisme de mise en question d'authentification anonyme (spécifié au paragraphe 22.1 de la [RFC3261]).

Il est fortement RECOMMANDÉ d'inclure l'IDr dans le message de l'initiateur. Si il n'est pas inclus, son absence peut être utilisée pour des besoins de déni de service (le plus grand impact étant sur les méthodes de clé publique et DH) où un message destiné à d'autres entités est envoyé à la cible. En fait, la cible peut vérifier correctement la signature du fait que l'identifiant de l'initiateur est correct et que le message est en fait signé par l'initiateur prétendu (par exemple, en redirigeant le trafic à partir d'une autre session).

Cependant, dans la méthode de la clé publique, la clé enveloppe et le MAC vont assurer que le message n'est pas accepté (mais, par rapport à un message falsifié normal, où la vérification de signature détecterait le problème, un déchiffrement de clé publique supplémentaire est nécessaire pour détecter le problème dans ce cas).

Dans la méthode DH, un message serait accepté (sans détecter l'erreur) et une réponse (et un état) serait créée pour la demande malveillante.

Comme on l'a aussi exposé au paragraphe 5.4, le compromis entre synchronisation horaire et taille de l'antémémoire de répétition peut être affecté dans le cas, par exemple, d'une attaque de DoS invasive. Cependant, si on suit la recommandation d'utiliser l'ajustement dynamique de la taille de l'antémémoire de répétition, on pense que le client ne sera pas dans la plupart des cas en mesure de gérer l'antémémoire de répétition. Bien sûr, comme la taille de l'antémémoire de répétition diminue, la synchronisation horaire requise est plus restreinte. Cependant, un plus gros problème durant une telle attaque serait probablement de traiter le messages (par exemple, vérifier les signatures/MAC) à cause de la charge de calcul que cela implique.

9.6 Établissement de session

On devrait noter que si le protocole d'établissement de session n'est pas sûr, il peut y avoir contre lui des attaques qui vont avoir des implications de sécurité indirectes sur les flux de support sécurisés. Cela ne s'applique cependant qu'aux groupes (et n'est pas spécifique de MIKEY). La menace est qu'un membre du groupe peut rediriger un flux d'un membre du groupe à un autre. Cela va avoir les mêmes implications que quand un membre essaye de se faire passer pour un autre membre, par exemple, en changeant son adresse IP. Si cela pose un problème, il est RECOMMANDÉ qu'un schéma d'authentification d'origine des données (DOA, *Data Origin Authentication*) (par exemple, des signatures numériques) soit appliqué au protocole de sécurité.

Rediriger un flux peut bien sûr être fait même si ce n'est pas un groupe. Cependant, l'effet ne sera pas le même que dans un groupe où l'usurpation d'identité peut être faite si DOA n'est pas utilisé. Là, la redirection va seulement dénier au receveur la possibilité de recevoir (ou va juste retarder) les données.

10. Considérations relatives à l'IANA

Le présent document définit plusieurs nouveaux espaces de noms associés aux charges utiles MIKEY. Cette section résume les espaces de noms pour lesquels il est demandé à l'IANA de gérer l'allocation des valeurs. Il est demandé à l'IANA d'enregistrer les valeurs prédéfinies dans les paragraphes consacrés à chaque espace de noms. Il est aussi demandé à l'IANA de gérer la définition des valeurs qui seront ajoutées à l'avenir. Sauf mention contraire explicite, les valeurs dans la gamme de 0 à 240 pour chaque espace de noms DEVRAIENT être approuvées par le processus de consensus de l'IETF et les valeurs dans la gamme de 241 à 255 sont réservées pour utilisation privée, selon la [RFC2434].

Il est demande que les espaces de noms pour les champs suivants dans la charge utile En-tête commun (du paragraphe 6.1) soient gérés par l'IANA (entre parenthèses sont les référence au tableau des valeurs initialement enregistrées) :
* Version
* Type de données (Tableau 6.1.a)
* Prochaine charge utile (Tableau 6.1.b)
* Fonction PRF (Tableau 6.1.c). Cet espace de noms est entre 0 et 127, et les valeurs entre 0 et 111 devraient être approuvées par le processus de consensus de l'IETF et les valeurs entre 112 et 127 sont réservées pour usage privé.
* Type trsp ID CS (Tableau 6.1.d)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Transport de données de clé ((au paragraphe 6.2) soient gérés par l'IANA :
* Algorithme de chiffrement (Tableau 6.2.a)
* Algorithme de MAC (Tableau 6.2.b)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Données d'enveloppe (du paragraphe 6.3) soient gérés par l'IANA :
* C (Tableau 6.3)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Données DH (du paragraphe 6.4) soient gérés par l'IANA :
* DH-Group (Tableau 6.4)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Signature (du paragraphe 6.5) soient gérés par l'IANA :
* Type de S (Tableau 6.5)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Horodatage (du paragraphe 6.6) soient gérés par l'IANA :
* Type de TS (Tableau 6.6)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile ID et Certificat (du paragraphe 6.7) soient gérés par l'IANA :
* Type d'ID (Tableau 6.7.a)
* Type de Cert (Tableau 6.7.b)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Hachage de certificat (du paragraphe 6.8) soient gérés par l'IANA :
* Fonction de hachage (Tableau 6.8)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Politique de sécurité (du paragraphe 6.10) soient gérés par l'IANA :
* Type de protocole (Tableau 6.10)

Pour chaque protocole de sécurité qui utilise MIKEY, un ensemble unique de paramètres PEUT être enregistré.
D'après le paragraphe 6.10.1.
* Type SRTP (Tableau 6.10.1.a)
* Algorithme de chiffrement SRTP (Tableau 6.10.1.b)
* Algorithme d'authentification SRTP (Tableau 6.10.1.c)
* PRF SRTP (Tableau 6.10.1.d)
* Ordre de FEC (Tableau 6.10.1.e)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Erreur (d'après le paragraphe 6.12) soient gérés par l'IANA :

* N° d'erreur (Tableau 6.12)

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Données de clé (d'après le paragraphe 6.13) soient gérés par l'IANA :
* Type (Tableau 6.13.a). Cet espace de noms est entre 0 et 16, et les valeurs devraient être approuvées par le processus de consensus de l'IETF.
* KV (Tableau 6.13.b). Cet espace de noms est entre 0 et 16, et les valeurs devraient être approuvées par le processus de consensus de l'IETF.

Il est demandé que les espaces de noms pour les champs suivants dans la charge utile Extensions générales (d'après le paragraphe 6.15) soient gérés par l'IANA :
* Type (Tableau 6.15).

10.1 Enregistrement MIME

Ce paragraphe donne pour instruction à l'IANA d'enregistrer le type de support MIME application/mikey. Cet enregistrement est comme suit :

Nom de type de support MIME : application
Nom de sous-type MIME : mikey
Paramètres requis : aucun
Paramètres facultatifs : version
version : numéro de version MIKEY du message inclus (par exemple, 1). S'il est absent, la version par défaut est 1.
Considérations de codage : binaire, codé en base64
Considérations de sécurité : voir la section 9 du présent mémoire
Considérations d'interopérabilité : aucune
Spécification publiée : le présent mémoire

11. Remerciements

Les auteurs tiennent à remercier Mark Baugher, Ran Canetti, Martin Euchner, Steffen Fries, Peter Barany, Russ Housley, Pasi Ahonen (avec son groupe) Rolf Blom, Magnus Westerlund, Johan Bilien, Jon-Olov Vatn, Erik Eliasson, et Gerhard Strangar pou leurs précieuses réactions sur ce document.

12 Références

12.1 Références normatives

[PSS] PKCS #1 v2.1 - RSA Cryptography Standard, RSA Laboratories, 14 juin 2002, www.rsalabs.com

[RFC2104] H. Krawczyk, M. Bellare et R. Canetti, "HMAC : Hachage de clés pour l'authentification de message", février 1997.

[RFC2119] S. Bradner, "Mots clés à utiliser dans les RFC pour indiquer les niveaux d'exigence", BCP 14, mars 1997.

[RFC2396] T. Berners-Lee, R. Fielding et L. Masinter, "Identifiants de ressource uniformes (URI) : Syntaxe générique", août 1998. (*Obsolète, voir* RFC3986)

[RFC2412] H. Orman, "Protocole OAKLEY de détermination de clés", novembre 1998. *(Information)*

[RFC2434] T. Narten et H. Alvestrand, "Lignes directrices pour la rédaction d'une section Considérations relatives à l'IANA dans les RFC", BCP 26, octobre, 1998. *(Rendue obsolète par la* RFC5226*)*

[RFC2486] B. Aboba, M. Beadles, "Identifiant d'accès réseau", janvier 1999. (*Obsolète, voir* RFC4282) *(P.S.)*

[RFC3280] R. Housley, W. Polk, W. Ford et D. Solo, "Profil de certificat d'infrastructure de clé publique X.509 et de liste de révocation de certificat (CRL) pour l'Internet", avril 2002. *(Obsolète, voir* RFC5280*)*

[RFC3394] J. Schaad, R. Housley, "Algorithme d'enveloppe de clés pour la norme de chiffrement évoluée (AES)", septembre 2002. *(Information)*

[RFC3711] M. Baugher et autres, "Protocole de transport sécurisé en temps réel (SRTP)", mars 2004.

[SHA-1] NIST, FIPS PUB 180-1, "Secure Hash Standard", avril 1995.

12.2 Références pour information

[AKE] Canetti, R. et H. Krawczyk, "Analysis of Key-Exchange Protocols et their use for Building Secure Channels", Eurocrypt 2001, LNCS 2054, pp. 453-474, 2001.

[BDJR] Bellare, M., Desai, A., Jokipii, E., et P. Rogaway, "A Concrete Analysis of Symmetric Encryption: Analysis of the DES Modes of Operation", dans Proceedings of the 38th Symposium on Foundations of Computer Science, IEEE, 1997, pp. 394-403.

[BMGL] Hastad, J. et M. Naslund: "Practical Construction et Analysis of Pseduo-randomness Primitives", Proceedings of Asiacrypt 2001, LNCS. vol 2248, pp. 442-459, 2001.

[DBJ] Johnson, D.B., "Theoretical Security Concerns with TLS use of MD5", Contribution au groupe de travail ANSI X9F1, 2001.

[FIPS] "Security Requirements for Cryptographic Modules", Federal Information Processing Standard Publications (FIPS PUBS) 140-2, décembre 2002.

[HAC] Menezes, A., van Oorschot, P., et S. Vanstone, "Handbook of Applied Cryptography", CRC press, 1996.

[ISO1] ISO/CEI 9798-3: 1997, "Technologies de l'information – techniques de sécurité – authentification d'entité – partie 3 : mécanismes utilisant des techniques de signature numérique".

[ISO2] ISO/CEI 11770-3: 1997, "Technologies de l'information – techniques de sécurité – gestion de clé – partie 3 : mécanismes utilisant des techniques de signature numérique".

[ISO3] ISO/CEI 18014 "Technologies de l'information – techniques de sécurité – services d'horodatage, Partie 1-3".

[LOA] Burrows, Abadi, et Needham, "A logic of authentication", ACM Transactions on Computer Systems 8 No.1 (février 1990), 18-36.

[LV] Lenstra, A. K. et E. R. Verheul, "Suggesting Key Sizes for Cryptosystems", http://www.cryptosavvy.com/suggestions.htm

[RFC1305] D. Mills, "Protocole de l'heure du réseau, version 3, spécification, mise en œuvre et analyse", STD 12, mars 1992. *(Remplacée par* RFC5905*)*

[RFC1750] D. Eastlake, 3rd et autres, "Recommandations d'aléa pour la sécurité", décembre 1994. *(Info., remplacée par la* RFC4086)

[RFC2246] T. Dierks et C. Allen, "Protocole TLS version 1.0", janvier 1999.

[RFC2326] H. Schulzrinne, A. Rao et R. Lanphier, "Protocole de flux directs en temps réel (RTSP)", avril 1998.

[RFC2327] M. Handley et V. Jacobson, "SDP : Protocole de description de session", avril 1998. (*Obsolète; voir* RFC4566)

[RFC2409] D. Harkins et D. Carrel, "L'échange de clés Internet (IKE)", novembre 1998. (*Obsolète, voir la* RFC4306*)*

[RFC2560] M. Myers, R. Ankney, A. Malpani, S. Galperin et C. Adams, "Protocole d'état de certificat en ligne d'infrastructure de clé publique X.509 pour l'Internet - OCSP", juin 1999. *(P.S.)*

[RFC3261] J. Rosenberg et autres, "SIP : Protocole d'initialisation de session", juin 2002. (*Mise à jour par* RFC3265, RFC3853, RFC4320, RFC4916, RFC5393, RFC6665))

[RFC3547] M. Baugher, B. Weis, T. Hardjono et H. Harney, "Le domaine d'interprétation de groupe", juillet 2003. *(Obsolète, voir la* RFC6407*)*

[RFC4046] M. Baugher et autres, "Architecture de gestion de clé de groupe de diffusion groupée sécurisée (MSEC)", avril 2005. *(Info.)*

[RFC4535] H. Harney et autres, "GSAKMP : protocole de gestion de clés d'association de groupe sécurisé", juin 2006. *(P.S.)*

[RFC4567] J. Arkko et autres, "Extensions de gestion de clés pour le protocole de description de session (SDP) et le protocole d'écoulement en temps réel (RTSP)", juillet 2006. *(P.S.)*

[SHA256] NIST, "Description of SHA-256, SHA-384, et SHA-512", http://csrc.nist.gov/encryption/shs/sha256-384-512.pdf

Appendice A Relation MIKEY - SRTP

La terminologie de MIKEY diffère de celle utilisée dans SRTP car MIKEY a besoin d'être plus général, et n'est pas lié au seul SRTP. Donc, il pourrait être difficle de voir les relations entre les clés et paramètres générés dans MIKEY et ceux utilisés par SRTP. Cette section donne quelques indications sur leurs relations.

MIKEY	**SRTP**
Session de chiffrement	flux SRTP (normalement avec le flux SRTCP en rapport)
SA de données	entrée au contexte cryptographique de SRTP
TEK	clé maîtresse SRTP

La SA de données est construite par une TEK et la politique de sécurité échangée. SRTP peut utiliser un MKI pour indexer la TEK ou TGK (la TEK est alors déduite de la TGK qui est associée au MKI correspondant), voir ci-dessous.

A.1 Interactions MIKEY-SRTP

Dans ce qui suit, on donne un bref résumé de l'interface entre SRTP et MIKEY et du traitement qui a lieu. On décrit seulement le côté receveur SRTP, le côté envoyeur exigera une interface analogue.

1. Lorsqu'un paquet SRTP arrive chez le receveur et y est traité, le triplet <SSRC, adresse de destination, accès de destination> est extrait du paquet et utilisé pour restituer le contexte cryptographique SRTP correct, et donc, la SA de données. (La restitution réelle peut, par exemple, être faite par une demande explicite de la mise en œuvre SRTP à MIKEY, ou par l'accès de la mise en œuvre SRTP à une "base de données", tenue par MIKEY. L'application va normalement décider quelle mise en œuvre est préférée.)

2. Si un MKI est présent dans le paquet SRTP, il est utilisé pour pointer sur la clé correcte au sein de la SA. Autrement, si le dispositif <From, To> de SRTP est utilisé, le ROC||SEQ du paquet est utilisé pour déterminer la clé correcte.

3. Selon que la clé envoyée dans MIKEY (obtenue à l'étape 2) est une TEK ou une TGK, il y a maintenant deux cas.
   - Si la clé obtenue à l'étape 2 est la TEK elle-même, elle est utilisée directement par SRTP comme clé maîtresse.
   - Si la clé est une TGK, la transposition avec le CS_ID (interne à MIKEY, § 6.1.1) permet à MIKEY de calculer la TEK correcte à partir de la TGK, comme décrit au paragraphe 4.1, avant que SRTP l'utilise.

Si plusieurs TGK (ou TEK) sont envoyées, il est RECOMMANDÉ que chaque TGK (ou TEK) soit associée à un MKI distinct. Il est RECOMMANDÉ que l'utilisation de <From, To> dans ce scénario soit limitée aux cas très simples, par exemple, un seul flux.

À côté de la clé maîtresse réelle, d'autres informations de la SA de données (par exemple, les identifiants de transformation) seront bien sûr aussi communiquées de MIKEY à SRTP.

Adresse des auteurs

Jari Arkko
Ericsson Research
02420 Jorvas
Finland
téléphone : +358 40 5079256
mél : jari.arkko@ericsson.com

Elisabetta Carrara
Ericsson Research
SE-16480 Stockholm
Sweden
téléphone : +46 8 50877040
mél : elisabetta.carrara@ericsson.com

Fredrik Lindholm
Ericsson Research
SE-16480 Stockholm
Sweden
téléphone : +46 8 58531705
mél : fredrik.lindholm@ericsson.com

Mats Naslund
Ericsson Research
SE-16480 Stockholm
Sweden
téléphone : +46 8 58533739
mél : mats.naslund@ericsson.com

Karl Norrman
Ericsson Research
SE-16480 Stockholm
Sweden
téléphone : +46 8 4044502
mél : karl.norrman@ericsson.com

Déclaration complète de droits de reproduction

Remerciement
Le financement de la fonction d'édition des RFC est actuellement assuré par la Internet Society.