UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T H.235

(11/2000)

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

SÉRIE H: SYSTÈMES AUDIOVISUELS ET MULTIMÉDIAS

Infrastructure des services audiovisuels – Aspects système

Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)

Recommandation UIT-T H.235

(Antérieurement Recommandation du CCITT)

RECOMMANDATIONS UIT-T DE LA SÉRIE H

SYSTÈMES AUDIOVISUELS ET MULTIMÉDIAS

CARACTÉRISTIQUES DES SYSTÈMES VISIOPHONIQUES	H.100–H.199
INFRASTRUCTURE DES SERVICES AUDIOVISUELS	
Généralités	H.200–H.219
Multiplexage et synchronisation en transmission	H.220–H.229
Aspects système	**H.230–H.239**
Procédures de communication	H.240–H.259
Codage des images vidéo animées	H.260–H.279
Aspects liés aux systèmes	H.280–H.299
SYSTÈMES ET ÉQUIPEMENTS TERMINAUX POUR LES SERVICES AUDIOVISUELS	H.300–H.399
SERVICES COMPLÉMENTAIRES EN MULTIMÉDIA	H.450–H.499

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

Recommandation UIT-T H.235

Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)

Résumé

La présente Recommandation décrit des améliorations apportées dans le cadre de la série des Recommandations H.3xx afin d'y introduire des services de sécurité tels que *l'authentification* et le *secret des communications* (cryptage des données). Le procédé qui est proposé est applicable aussi bien aux simples conférences point à point qu'aux conférences point à multipoint, à partir de tous les terminaux faisant appel au protocole de commande décrit dans l'UIT-T H.245.

Par exemple, les systèmes H.323 fonctionnent sur des réseaux en mode paquet qui n'offrent pas une qualité de service garantie. La sûreté et la qualité du service offert par le réseau de base sont absentes pour les mêmes raisons techniques. Des communications sûres et en temps réel sur des réseaux non sûrs soulèvent généralement deux grands types de préoccupation: *l'authentification* et le *secret des communications*.

La présente Recommandation décrit l'infrastructure de sécurité et les techniques spécifiques de secret des communications que les terminaux multimédias conformes à la série H.3xx doivent utiliser. Elle traite les questions relatives aux conférences interactives, c'est-à-dire, entre autres domaines, l'authentification et le secret des communications de tous les flux médias échangés en temps réel au cours d'une conférence. Elle indique le protocole et les algorithmes nécessaires entre les entités H.323.

La présente Recommandation fait appel aux capacités générales qui sont décrites dans l'UIT-T H.245: toute norme d'exploitation liée à ce protocole de commande pourra donc utiliser ce cadre de sécurité. L'on prévoit que, dans la mesure du possible, d'autres terminaux selon la série H pourront interfonctionner et utiliser directement les méthodes décrites ci-après. Dans un premier temps, la présente Recommandation n'assurera pas une mise en œuvre complète dans tous les domaines. Elle développera spécifiquement l'authentification des points d'extrémité et le secret des communications multimédias.

La présente Recommandation prévoit la possibilité de négocier les services et les capacités de façon générique. Elle prévoit également la possibilité de sélectionner les techniques et capacités cryptographiques utilisées. Leur mode d'emploi particulier dépend des capacités des systèmes, des exigences d'application et des contraintes propres aux politiques de sécurité. La présente Recommandation prend en compte divers algorithmes cryptographiques, avec diverses options appropriées à différents objectifs, comme les longueurs des clés. Certains algorithmes cryptographiques peuvent être attribués à des services de sécurité spécifiques (par exemple un algorithme pour un chiffrement rapide du flux média et un autre pour le codage de la signalisation).

Il convient également de noter que certains des algorithmes ou mécanismes cryptographiques dont on dispose pourront être réservés à l'exportation ou à d'autres fins nationales (par exemple avec des clés de longueur soumise à contrainte). La présente Recommandation prend en compte la signalisation d'algorithmes notoires, en plus de celle d'algorithmes cryptographiques non normalisés ou privatifs. Aucun algorithme n'est spécifiquement prescrit mais il est fortement conseillé que les points d'extrémité prennent en charge autant d'algorithmes applicables que possible afin de réaliser l'interopérabilité. Ce conseil est à rapprocher de l'idée que la conformité à l'UIT-T H.245 ne garantit pas l'interopérabilité de deux codecs d'entité.

La présente version de l'UIT-T H.235, qui remplace la première, contient de nombreuses améliorations telles que la cryptographie à courbe elliptique, des profils de sécurité (de type à simple mot de passe ou à signature numérique perfectionnée), de nouvelles contre-mesures de sécurité (protection contre l'inondation du média), la prise en charge de l'algorithme de cryptage avancé (AES) et du service d'extrémité; elle définit des identificateurs d'objet et introduit des modifications tirées du guide à l'usage des responsables de l'implémentation de l'UIT-T H.323.

Source

La Recommandation H.235 de l'UIT-T, révisée par la Commission d'études 16 (2001-2004) de l'UIT-T, a été approuvée le 17 novembre 2000 selon la procédure définie dans la Résolution 1 de l'AMNT.

La première version de l'UIT-T H.235 avait été approuvée par la Commission 16 de l'UIT-T le 6 février 1998.

Mots clés

Authentification, certificat, cryptage, gestion de clés, intégrité, profil de sécurité, sécurité multimédia, signature numérique.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

TABLE DES MATIÈRES

Recommandation UIT-T H.235

Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)

1 Domaine d'application

L'objectif principal de la présente Recommandation est d'assurer l'authentification, le secret des communications et l'intégrité dans le cadre du protocole actuel de la série H. Le texte de la présente Recommandation (2000) donne des détails sur l'implémentation avec le protocole H.323. On prévoit que ce cadre fonctionnera en liaison avec d'autres protocoles de la série H utilisant le protocole de commande H.245.

Les objectifs complémentaires de la présente Recommandation sont les suivants:

1) il y a lieu de développer une architecture de sécurité sous la forme d'un cadre extensible et flexible, permettant de mettre en œuvre un système de sécurité pour terminaux conformes à la série H. Ce cadre devra être fourni au moyen des capacités offertes par des services flexibles et indépendants, telles que la capacité de négocier et de sélectionner les techniques cryptographiques utilisées, ainsi que la façon de les utiliser;

2) assurer la sécurité de toutes les communications résultant de l'utilisation du protocole H.3xx, ce qui implique les questions d'établissement des connexions, de commande d'appel et d'échange de médias entre toutes les entités. Cette exigence comporte l'emploi de communications confidentielles (capacité de secret des communications) où l'on peut exploiter des fonctions d'authentification d'homologue ainsi que de protection de l'environnement de l'utilisateur contre les agressions qu'il pourrait subir;

3) la présente Recommandation ne doit pas interdire l'intégration d'autres fonctions de sécurité dans des entités H.3xx, pouvant les protéger contre des agressions issues du réseau;

4) la présente Recommandation ne doit pas limiter l'échelonnement de quelconques terminaux de la série de Recommandations H.3xx, selon les nécessités. Il peut s'agir aussi bien du nombre d'utilisateurs protégés que des niveaux de sécurité procurés;

5) le cas échéant, tous les mécanismes et toutes les capacités doivent être fournis indépendamment des couches ou topologies de transport sous-jacentes. D'autres moyens, hors du domaine d'application de la présente Recommandation, peuvent être requis pour contrer les menaces de ce type;

6) des dispositions doivent être prises pour le fonctionnement en environnement mixte (entités protégées et non protégées);

7) la présente Recommandation doit offrir la possibilité de distribuer des clés de session associées à la méthode cryptographique utilisée. (Ce qui n'implique pas que la gestion de clés publiques fondées sur des certificats doive faire partie de la présente Recommandation.)

8) la présente Recommandation propose deux profils de sécurité qui facilitent l'interopérabilité, l'un simple, mais sûr, de type à mot de passe (voir l'Annexe D), l'autre de type à signature utilisant des signatures numériques, des certificats et une infrastructure à clé publique (voir l'Annexe E), qui n'est pas sujet aux limitations du profil de l'Annexe D.

L'architecture de sécurité décrite dans la présente Recommandation ne part pas du principe que les participants se connaissent déjà. Elle suppose cependant que des précautions appropriées ont été prises pour protéger physiquement les points d'extrémité conformes à la série H. Le principal risque pour les communications est donc supposé être une indiscrétion dans le réseau ou une autre méthode de détournement de flux médias.

L'UIT-T H.323 donne la possibilité de conduire une conférence en mode audio, vidéo ou données entre plusieurs correspondants; mais elle ne donne pas à chaque participant la possibilité d'authentifier l'identité des autres participants. Elle ne permet pas non plus de privatiser les communications (c'est-à-dire de coder les flux).

Les terminaux de type UIT-T H.323, UIT-T H.324 et UIT-T H.310 font appel aux procédures de signalisation par canal logique selon l'UIT-T H.245, dans laquelle le contenu de chaque canal logique est décrit dès son ouverture. Des procédures sont prévues pour exprimer les capacités du récepteur et de l'émetteur. Les transmissions sont limitées à ce que les récepteurs peuvent décoder et ces derniers peuvent demander aux émetteurs un mode préférentiel particulier. Les capacités de sécurité de chaque entité terminale sont communiquées de la même façon que toutes les autres capacités de communication.

Certains terminaux de la série H (H.323) peuvent être utilisés en configuration multipoint. Le mécanisme de sécurité décrit dans la présente Recommandation permettra un fonctionnement sûr dans les environnements mettant en œuvre une exploitation par ponts de conférence (MCU) aussi bien centralisés que décentralisés.

2 Références normatives

La présente Recommandation se réfère à certaines dispositions des Recommandations UIT-T et textes suivants qui, de ce fait, en sont partie intégrante. Les versions indiquées étaient en vigueur au moment de la publication de la présente Recommandation. Toute Recommandation ou tout texte étant sujet à révision, les utilisateurs de la présente Recommandation sont invités à se reporter, si possible, aux versions les plus récentes des références normatives suivantes. La liste des Recommandations de l'UIT-T en vigueur est régulièrement publiée.

– UIT-T H.225.0 (2000), *Protocoles de signalisation d'appel et mise en paquets des trains multimédias dans les systèmes de communication multimédia en mode paquet*.

– UIT-T H.235 (1998), *Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)*.

– UIT-T H.245 (2000), *Protocole de commande pour communications multimédias*.

– UIT-T H.323 (2000), *Systèmes de communication multimédia en mode paquet*.

– UIT-T H.323 Annexe J (2000), *Fonctions de sécurité pour les systèmes H.323 Annexe F*.

– UIT-T Q.931 (1998), *Spécification de la couche 3 de l'interface utilisateur-réseau RNIS pour la commande de l'appel de base*.

– UIT-T X.509 (2000) | ISO/CEI 9594-8:2001, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: cadre général des certificats de clé publique et d'attribut*.

– UIT-T X.800 (1991), *Architecture de sécurité pour l'interconnexion en systèmes ouverts d'applications du CCITT*.

  ISO 7498-2:1989, *Systèmes de traitement de l'information – Interconnexion de systèmes ouverts – Modèle de référence de base – Partie 2: Architecture de sécurité*.

– UIT-T X.803 (1994) | ISO/CEI 10745:1995, *Technologies de l'information – Interconnexion des systèmes ouverts – Modèle de sécurité pour les couches supérieures*.

– UIT-T X.810 (1995) | ISO/CEI 10181-1:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: aperçu général.*

– UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: cadre d'authentification*.

– ISO/CEI 9797:1994, *Technologies de l'information – Techniques de sécurité – Mécanisme d'intégrité des données utilisant une fonction de contrôle cryptographique employant un algorithme de chiffrement par bloc*.

– ISO/CEI 9798-2:1999, *Technologies de l'information – Techniques de sécurité – Authentification d'entité – Partie 2: Mécanismes utilisant des algorithmes de chiffrement symétriques*.

– ISO/CEI 9798-3:1998, *Technologies de l'information – Techniques de sécurité – Authentification d'entité – Partie 3: Mécanismes utilisant des techniques de signature numériques*.

– ISO/CEI 9798-4:1999, *Technologies de l'information – Techniques de sécurité – Authentification d'entité – Partie 4: Mécanismes utilisant une fonction cryptographique de vérification*.

– ISO/CEI FCD 15946-1, *Technologies de l'information – Techniques de sécurité – Techniques cryptographiques basées sur les courbes elliptiques, Partie 1: Généralités*.

– ISO/CEI FCD 15946-2, *Technologies de l'information – Techniques de sécurité – Techniques cryptographiques basées sur les courbes elliptiques, Partie 2: Signatures digitales*.

– ATM Forum: af-sec-0100.002 (2001), *ATM Security Specification Version 1*.

– IETF RFC 1321 (1992), *The MD5 Message-Digest Algorithm*.

– IETF RFC 2104 (1997), *HMAC: Keyed-Hashing for Message Authentication* (Adressage dispersé sur clés calculées pour authentification de messages)

– IETF RFC 2138 (1997), *Remote Authentication Dial In User Service (RADIUS)*.

– IETF RFC 2246 (1999), *The TLS Protocol Version 1.0*.

– IETF RFC 2401 (1998), *Security Architecture for the Internet Protocol* (Architecture de sécurité pour le protocole Internet).

– IETF RFC 2402 (1998), *IP Authentication Header*.

– IETF RFC 2407 (1998), *The Internet IP Security Domain of Interpretation for ISAKMP*.

– IETF RFC 2412 (1998), *The OAKLEY Key Determination Protocol*.

– IETF RFC 2437 (1998), *PKCS #1: RSA Encryption Version 2.0*.

– IETF RFC 2459 (1999), *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*.

3 Termes et définitions

Pour les besoins de la présente Recommandation, les définitions figurant au paragraphe 3/H.323, au paragraphe 3/H.225.0 et au paragraphe 3/H.245 s'appliquent, en plus de celles que contient le présent paragraphe. Certains des termes suivants sont utilisés selon la définition donnée dans l'UIT-T X.800 | ISO 7498-2 et dans l'UIT-T X.803, l'UIT-T X.810 et l'UIT-T X.811.

3.1 contrôle d'accès: précaution prise contre l'utilisation non autorisée d'une ressource, y compris l'utilisation d'une ressource d'une façon non autorisée (X.800).

3.2 authentification: attestation de l'identité revendiquée par une entité (X.811).

3.3 autorisation: octroi d'une permission sur la base d'une identité authentifiée.

3.4 attaque: activités entreprises pour contourner ou exploiter des déficiences constatées dans les mécanismes de sécurité d'un système. Une attaque directe d'un système exploite des déficiences dans les algorithmes, principes ou propriétés sous-tendant un mécanisme de sécurité. Les attaques indirectes consistent à contourner le mécanisme ou à en provoquer une utilisation incorrecte par le système.

3.5 certificat: ensemble de données relatives à la sécurité, émis par une autorité de sécurité ou par un tiers de confiance en même temps que des informations de sécurité qui sont utilisées pour fournir les services d'intégrité et d'authentification d'origine des données (X.810). Dans la présente Recommandation, ce terme vise des certificats "à clé publique" qui sont des valeurs représentant une clé publique de détenteur (et d'autres informations facultatives), ces valeurs ayant été vérifiées et signées par une autorité de confiance sous une forme infalsifiable.

3.6 chiffre: algorithme cryptographique ou transformée mathématique.

3.7 confidentialité: caractéristique qui empêche la divulgation des informations à des individus, entités ou processus non autorisés.

3.8 algorithme cryptographique: fonction mathématique qui calcule un résultat à partir d'une ou de plusieurs valeurs d'entrée.

3.9 chiffrement; cryptage: processus consistant à rendre des données illisibles par des entités non autorisées après application d'un algorithme cryptographique (ou de cryptage). Le déchiffrement (décryptage) est l'opération inverse par laquelle le texte chiffré est transformé en texte clair.

3.10 intégrité: caractéristique de données qui n'ont pas été altérées de façon non autorisée.

3.11 gestion de clés: production, stockage, distribution, suppression, archivage et application de clés conformément à une politique de sécurité (X.800).

3.12 flux média: flux audio, vidéo ou de données, ou combinaison quelconque de ces types de flux. Les flux médias acheminent des données d'utilisateur ou d'application (capacité utile) mais pas de données de commande.

3.13 non-répudiation: protection contre le déni, par une des entités impliquées dans une communication, d'avoir participé à tout ou partie de celle-ci.

3.14 secret des communications: mode de communication dans lequel seules les parties explicitement habilitées peuvent interpréter la communication. Le secret des communications est normalement réalisé par cryptage et par partage de clé(s) pour accéder au chiffre.

3.15 canal privé: dans la présente Recommandation, un canal privé est celui qui résulte d'une négociation préalable par canal sécurisé et qui peut servir à acheminer des flux médias.

3.16 cryptographie à clés publiques: système de cryptage qui fait appel (pour le cryptage et le décryptage) à des clés asymétriques liées par une relation mathématique qui ne peut logiquement pas être calculée.

3.17 profil de sécurité: (sous-) ensemble cohérent de procédures et caractéristiques interopérables, tirées de l'UIT-T H.235, très utiles pour sécuriser des communications multimédias H.323 entre des entités concernées dans un scénario donné.

3.18 submersion: agression visant la fonction de refus de service d'un système par l'envoi, à celui-ci, d'un grand nombre de données non autorisées. Un cas particulier est la submersion d'un média par l'envoi de paquets RTP à des ports UDP. Généralement, le système est submergé de paquets et le traitement correspondant nécessite de précieuses ressources.

3.19 algorithme cryptographique symétrique (à clés secrètes): algorithme permettant de réaliser le chiffrement ou le déchiffrement correspondant, dans lequel la même clé est requise à la fois pour le chiffrement et pour le déchiffrement (X.810).

3.20 menace: violation potentielle de la sécurité (X.800).

4 Symboles et abréviations

La présente Recommandation utilise les abréviations suivantes:

CRL	liste de révocations de certificat (*certificate revocation list*)
DSS	norme de signature numérique (*digital signature standard*)
ECC et EC	système de cryptage à courbe elliptique (voir la section 8.7 de "*Section ATM Forum Security Specification*" *Version 1.1*)
EC-GDSA	signature numérique à courbe elliptique avec appendice analogue à l'algorithme de signature numérique NIST (DSA); (voir aussi le chapitre 5 de [ISO/CEI 15946 2])
ECKAS-DH	système de concordance de clés à courbe elliptique – Diffie-Hellman. Système de concordance de clés Diffie-Hellman utilisant la cryptographie à courbe elliptique
IPSEC	sécurité du protocole Internet (*Internet protocol security*)
QS	qualité de service
RSA	algorithme à clé publique de Rivest, Shamir et Adleman
SDU	unité de données de service (*service data unit*)
TLS	sécurité de la couche Transport (*transport level security*)

5 Conventions

Dans la présente Recommandation, les conventions suivantes s'appliquent:

– l'auxiliaire "doit/doivent" indique une prescription impérative;

– l'auxiliaire "devrait/devraient" indique une mesure suggérée mais facultative;

– l'auxiliaire "peut/peuvent" indique une possibilité d'action plutôt qu'une recommandation de résultat.

Sauf énumération explicite d'une autre Recommandation, les références aux paragraphes, sous-paragraphes, annexes et appendices se rapportent à ceux de la présente Recommandation. Par exemple, la référence "1.4" correspond au paragraphe 1.4 de la présente Recommandation. La référence "6.4/H.245" correspond au paragraphe 6.4 de la Recommandation H.245.

La présente Recommandation décrit l'utilisation de "n" types de message différents: H.245, RAS, Q.931, etc. Pour établir une distinction entre ces différents types de message, la convention suivante est utilisée: les messages et noms de paramètres H.245 se composent de plusieurs mots concaténés qui sont mis en valeur par un caractère gras (**maximumDelayJitter**); les noms de message RAS sont représentés par des abréviations à trois lettres (**ARQ**); les noms de message Q.931 se composent d'un ou de deux mots dont la première lettre est en majuscule (**Call Proceeding**).

6 Introduction au système

6.1 Résumé

1) le canal de signalisation d'appel peut être sécurisé au moyen du protocole TLS [TLS] ou IPSEC [IPSEC] à un accès dont la sûreté est bien établie (H.225.0);
2) les utilisateurs peuvent être authentifiés soit au cours de la connexion d'appel initiale, soit au cours du processus de sécurisation du canal H.245 et/ou par échange de certificats sur le canal H.245;
3) les capacités de cryptage d'un canal média sont déterminées par des extensions du mécanisme existant de négociation de capacité;
4) la distribution initiale par l'entité maîtresse des données relatives aux clés s'effectue par les messages H.245 **OpenLogicalChannel** ou **OpenLogicalChannelAck**;
5) la redéfinition des clés peut s'effectuer par les commandes H.245: **EncryptionUpdateRequest** et **EncryptionUpdate**;
6) la distribution des données relatives aux clés est protégée soit par l'exploitation du canal H.245 en tant que canal privé ou par protection spécifique des données de clé par échange des certificats sélectionnés;
7) les protocoles de sécurité présentés sont conformes soit à des normes ISO publiées ou à des normes proposées par le groupe IETF.

6.2 Authentification

Le processus d'authentification vérifie que ceux qui répondent sont bien ceux qu'ils disent être. L'authentification peut être réalisée dans le cadre de l'échange de certificats à clé publique ou dans celui d'un échange faisant appel à une information secrète, partagée entre les entités en cause. Il peut s'agir d'un mot de passe statique ou d'un autre type d'information arbitraire.

La présente Recommandation décrit le protocole d'échange des certificats mais ne spécifie pas les critères permettant de les vérifier et de les accepter les uns en fonction des autres. En général, les certificats donnent au vérificateur une certaine garantie que celui qui présente le certificat est la personne qu'il déclare être. L'échange de certificats a pour objet d'authentifier *l'utilisateur* du point d'extrémité et non simplement le point d'extrémité physique. L'utilisation de certificats numériques permet de prouver, par protocole d'authentification, que ceux qui répondent possèdent les clés privées correspondant aux clés publiques contenues dans les certificats. Cette authentification protège contre les agressions par entremetteur mais ne prouve pas automatiquement l'identité de ceux qui répondent. Pour cela, il faut normalement qu'une certaine politique s'applique au reste du contenu des certificats. Pour les certificats d'autorisation par exemple, le certificat contiendra normalement l'identification du fournisseur de service ainsi qu'une certaine identification du compte d'utilisateur, prescrite par le fournisseur de service.

Le cadre d'authentification présenté dans la présente Recommandation ne prescrit pas le contenu des certificats (c'est-à-dire qu'il ne spécifie pas de politique relative aux certificats) au-delà de ce qui est requis par le protocole d'authentification. Une application utilisant ce cadre pourra toutefois imposer des prescriptions politiques de haut niveau comme la présentation du certificat à l'utilisateur pour approbation. Cette politique de niveau supérieur pourra soit être automatisée au sein de l'application soit nécessiter une interaction humaine.

Pour l'authentification qui ne fait pas appel à des certificats numériques, la présente Recommandation indique la signalisation permettant de réaliser divers scénarios d'épreuve/réponse. Cette méthode d'authentification nécessite une coordination préalable entre les entités communicantes, de façon qu'un secret partagé soit obtenu. Un exemple de cette méthode serait celui d'un client abonné à un service.

Une troisième option permet de réaliser l'authentification dans le contexte d'un protocole de sécurité distinct tel que la sécurité TLS [TLS] ou IPSEC [IPSEC].

Des entités homologues peuvent prendre en charge une authentification aussi bien bidirectionnelle qu'unidirectionnelle. Cette authentification peut se produire sur tout ou partie des voies de communication.

Tous les mécanismes d'authentification spécifiques qui sont décrits dans la présente Recommandation sont identiques aux algorithmes mis au point par l'ISO (ou en sont dérivés), comme spécifié dans les Parties 2 à 3 de l'ISO/CEI 9798 ou sont fondés sur des protocoles IETF.

6.2.1 Certificats

La normalisation des certificats, y compris leur production, leur administration et leur distribution, est hors du domaine d'application de la présente Recommandation. Les certificats utilisés pour établir des canaux sûrs (signalisation d'appel et/ou commande d'appel) doivent être conformes aux prescriptions de tout protocole qui a été négocié pour sécuriser ces canaux.

Il y a lieu de noter que, pour l'authentification utilisant des certificats à clé publique, les points d'extrémité sont appelés à fournir des signatures numériques utilisant la valeur de clé privée associée. Le seul échange de certificats à clé publique ne suffit pas à protéger contre les agressions par entremetteur. Les protocoles H.235 sont conformes à cette exigence.

6.3 Sécurité lors de l'établissement d'appel

Il y a au moins deux raisons pour sécuriser le canal d'établissement d'appel (par exemple par message H.323 utilisant Q.931). La première est l'exécution d'une authentification simple, avant d'accepter l'appel. La deuxième vise à permettre une autorisation d'appel. Si cette fonction est souhaitée dans le terminal conforme à la série H, il y a lieu d'utiliser un mode de communication sécurisé (tel que TLS/IPSEC pour H.323) avant l'échange des messages de connexion d'appel. En variante, l'autorisation peut être donnée sur la base d'une authentification de service spécifique, dont les contraintes de politique sont hors du domaine d'application de la présente Recommandation.

6.4 Sécurité de la commande d'appel (H.245)

Le canal de commande d'appel (H.245) devrait également être sécurisé de quelque façon, afin d'offrir ensuite un média secret. Le canal H.245 doit être sécurisé par un quelconque mécanisme de secret des communications (dont la négociation comporte l'option "aucun"). Les messages H.245 sont utilisés pour signaler les algorithmes et clés de cryptage utilisés dans les canaux de médias partagés et privés. Cette capacité permet de crypter, canal logique par canal logique, différents canaux de média au moyen de différents mécanismes. Par exemple, lors de conférences multipoint centralisées, différentes clés peuvent être utilisées pour les différents flux destinés à chaque point d'extrémité. Cela permet de sécuriser les flux médias destinés à chaque point d'extrémité de la conférence. Pour utiliser les messages H.245 de manière sûre, l'ensemble du canal H.245 (canal logique 0) devrait être ouvert après sécurisation négociée.

Le mécanisme par lequel un canal H.245 est sécurisé dépend des terminaux série H utilisés. La seule exigence imposée à tous les systèmes utilisant cette structure de sécurité est que chacun d'eux possède une certaine capacité permettant de négocier et/ou de signaler que le canal H.245 doit être exploité d'une certaine manière sécurisée avant d'être effectivement initialisé. Par exemple, le protocole H.323 utilisera les messages de signalisation de connexion H.225.0 pour réaliser cette condition.

6.5 Secret des communications par flux médias

La présente Recommandation décrit le secret des communications multimédias pour des flux médias acheminés par transport en mode paquet. Ces canaux peuvent être unidirectionnels dans le cadre de la définition des canaux logiques H.245. Il n'est pas prescrit que ces canaux soient unidirectionnels dans la couche Physique ou Transport.

Une première étape pour réaliser le secret des communications multimédias devrait être la fourniture d'un canal de commande privé permettant d'établir des bases de construction de clés et/ou l'établissement des canaux logiques devant transporter les flux médias cryptés. A cette fin, lors du fonctionnement en conférence sécurisée, tout point d'extrémité participant peut utiliser un canal H.245 crypté. Cette procédure permet de protéger la sélection des algorithmes cryptographiques et les clés de cryptage transmises dans la commande H.245 **OpenLogicalChannel**.

Le canal H.245 sécurisé peut être exploité avec des caractéristiques différentes des canaux médias privés, dans la mesure où il procure un niveau de secret des communications acceptable par les deux parties. Il permet aux mécanismes de sécurité protégeant les flux médias et tous canaux de commande de fonctionner de manière totalement indépendante, en fournissant des niveaux de robustesse et de complexité totalement différents.

S'il est prescrit que le canal H.245 soit exploité de manière non cryptée, les clés spécifiques de cryptage de média peuvent être chiffrées séparément par les parties engagées, de la manière qui a été signalée et convenue. Un canal logique de type **h235Control** peut être utilisé pour fournir les données protégeant les clés de cryptage de média. Ce canal logique peut être exploité par tout mode négocié à cette fin.

Le secret (cryptage) des communications de données acheminées dans les canaux logiques doit avoir la forme spécifiée par la commande **OpenLogicalChannel**. Les informations d'en-tête spécifiques à la couche Transport ne doivent pas être chiffrées. Le secret des données doit être fondé sur un cryptage de bout en bout.

6.6 Eléments crédibilisés

La base de l'authentification (confiance) et du secret des communications est définie par les terminaux du canal de communication. Pour un canal d'établissement de connexion, ces terminaux peuvent être ceux de l'appelant et d'un élément du réseau d'accueil. Par exemple, un poste téléphonique "escompte" que le commutateur du réseau le connectera au poste dont le numéro a été composé. C'est pourquoi toute entité à laquelle aboutit un canal de commande H.245 crypté ou un canal logique de type **encryptedData** doit être considérée comme un élément crédibilisé de la connexion. Ces entités peuvent être des ponts de conférence ou des têtes de ligne (passerelles). Le résultat de la crédibilisation d'un élément est l'assurance de pouvoir révéler en confiance à cet élément le mécanisme de secret des communications (algorithme et clé).

Compte tenu de ce qui précède, il incombe aux participants du chemin de communication d'authentifier tout un chacun des éléments "crédibilisés". Pour cela, on procédera normalement à un échange de certificats comme dans le cas de l'authentification de bout en bout normale. La présente Recommandation ne prescrira aucun niveau spécifique d'authentification et se limitera à suggérer que ce niveau soit acceptable par toutes les entités faisant appel aux éléments crédibilisés. Les détails relatifs à un modèle et à une politique de certificats applicables à la crédibilisation feront l'objet d'un complément d'étude.

Le secret des communications ne peut être assuré entre les deux points d'extrémité que si les connexions entre éléments crédibilisés sont démontrées avoir été protégées contre les agressions par entremetteur.

6.6.1 Dépôt de clé

Bien que cela ne soit pas spécifiquement requis pour le fonctionnement, la présente Recommandation contient des dispositions pour conférer aux entités utilisant le protocole H.235 la capacité dite "tiers de confiance" (TTP, *trusted third party*) dans le cadre des éléments de signalisation.

La possibilité de récupérer les clés de codage de média perdues doit être prise en charge par les installations lorsqu'une telle capacité est souhaitable ou requise.

Le dépôt de clé est une fonctionnalité qui est souvent désignée par le terme "tiers de confiance" (TTP). Cette fonctionnalité reste à étudier.

6.7 Non-répudiation

A étudier.

7 Procédures d'établissement de connexion

7.1 Introduction

Comme indiqué dans le paragraphe introduction du système, aussi bien le canal de connexion d'appel (H.225.0 pour terminaux série H.323) que le canal de commande d'appel (H.245) doivent fonctionner dans le mode négocié, sécurisé ou non sécurisé, à partir du premier échange de messages. Pour le canal de connexion d'appel, le mode de sécurité est déterminé *a priori* [pour un terminal H.323, un point TSAP sécurisé par TLS (accès 1300) doit être utilisé pour les messages Q.931]. Pour le canal de commande d'appel, le mode de sécurité est déterminé par les informations transmises dans le protocole d'établissement de connexion initial, utilisé par le terminal série H.

Lorsqu'il n'y a pas de chevauchement entre capacités de sécurité, le terminal appelé peut refuser la connexion. L'erreur renvoyée ne devrait pas contenir de renseignements sur un quelconque défaut de concordance entre messages de sécurité. Le terminal appelant devra déterminer l'origine du problème par d'autres moyens. Lorsque le terminal appelant reçoit un message "d'acquittement de connexion" CONNECT ACKNOWLEDGE sans capacités de sécurité suffisantes, il y a lieu qu'il mette fin à l'appel.

Si les terminaux appelant et appelé sont des capacités de sécurité compatibles, chaque extrémité doit partir du principe que le canal H.245 doit fonctionner dans le mode sécurisé qui a été négocié. L'échec d'établissement du mode H.245 sécurisé qui est défini ici devrait être considéré comme une erreur de protocole et la connexion devrait être fermée.

8 Signalisation et procédures H.245

En général, les aspects relatifs au secret des communications par canaux médias sont commandés de la même façon que tout autre paramètre de codage: chaque terminal indique ses capacités, la source des données choisit un format à utiliser et le récepteur acquitte ou refuse le mode. Tous les aspects indépendants du mécanisme qui sont indépendants du transport, comme la sélection de l'algorithme, sont indiqués par des éléments génériques de canal logique. Les caractéristiques de transport telles que la synchronisation des algorithmes de clé ou de cryptage sont acheminées dans des structures propres à la couche Transport.

8.1 Fonctionnement avec canal H.245 sécurisé

En supposant que les procédures de connexion indiquées dans le paragraphe précédent (Procédures d'établissement de connexion) indiquent un mode de fonctionnement sécurisé, le dialogue négocié et l'authentification doivent être effectués pour le canal logique H.245 avant l'échange d'éventuels

autres messages H.245. S'il a été négocié, tout échange de certificats doit intervenir au moyen de tout mécanisme approprié pour les terminaux conformes à la série H. Après sécurisation du canal H.245, les terminaux utilisent le protocole H.245 comme ils le feraient en mode non sécurisé.

8.2 Fonctionnement avec canal H.245 non sécurisé

En variante, le canal H.245 peut fonctionner en mode non sécurisé et les deux entités ouvrent un canal logique sécurisé avec lequel l'authentification et le calcul du secret partagé sont effectués. Par exemple, une commande TLS ou IPSEC peut être utilisée par l'ouverture d'un canal logique dont le champ **dataType** contient une valeur pour le paramètre **h235Control**. Ce canal pourra ensuite être utilisé pour calculer un secret partagé protégeant d'éventuelles clés de session média ou pour transporter le message **EncryptionSync**.

8.3 Echange de capacités

Conformément aux procédures indiquées au 8.3/H.245 (Procédures d'échange de capacités) et conformément à la Recommandation de la série H applicable au système, les points d'extrémité échangent leurs capacités au moyen de messages H.245. Ces ensembles de capacités peuvent maintenant contenir des définitions indiquant des paramètres de sécurité et de cryptage. Par exemple, un point d'extrémité peut signaler des capacités d'émission et de réception de données vidéo H.261, normales ou cryptées.

Chaque algorithme de cryptage utilisé avec un codec média particulier implique une nouvelle définition de capacité. Comme pour toute autre capacité, les points d'extrémité peuvent indiquer, au cours de leur échange de capacité, des codecs cryptés aussi bien indépendants que dépendants. Cela permettra aux points d'extrémité de dimensionner leurs capacités de sécurité en fonction des charges et des ressources disponibles.

Une fois l'échange de capacités effectué, les points d'extrémité peuvent ouvrir des canaux logiques sécurisés pour médias de la même façon qu'ils le feraient en mode non sécurisé.

8.4 Rôle de maître

La relation maître-esclave H.245 est utilisée pour établir l'entité maîtresse en vue du fonctionnement en canal bidirectionnel et de la résolution d'autres conflits. Ce rôle de maître est également utilisé dans les méthodes de sécurité. Bien que le ou les modes de sécurité d'un flux média soient fixés par la source (en fonction des capacités du récepteur), le maître est le point d'extrémité qui produit la clé de cryptage. Cette production est effectuée sans tenir compte du fait que le maître est le récepteur ou l'émetteur du média crypté. Pour permettre le fonctionnement de canaux à destinations multiples avec clés partagées, le pont (qui est également le maître) doit normalement produire les clés.

8.5 Signalisation par canal logique

Les points d'extrémité ouvrent des canaux logiques pour médias en mode sécurisé de la même façon qu'ils le feraient pour des canaux logiques de médias en mode non sécurisé. Chaque canal peut fonctionner de manière totalement indépendante des autres canaux – en particulier pour ce qui est de la sécurité. Le mode particulier doit être défini dans le message **OpenLogicalChannel** du champ **dataType**. La clé de cryptage initiale doit être transmise dans le message **OpenLogicalChannel** ou **OpenLogicalChannelAck** selon la relation maître/esclave de l'expéditeur du message **OpenLogicalChannel**.

Le message **OpenLogicalChannelAck** doit faire fonction de confirmation du mode de cryptage. Si la commande **openLogicalChannel** n'est pas acceptable par le destinataire, le paramètre **dataTypeNotSupported** ou **dataTypeNotAvailable** (condition transitoire) doit être renvoyé dans le champ de cause du message **OpenLogicalChannelReject**.

Au cours de l'échange protocolaire qui établit le canal logique, la clé de cryptage doit être transmise du maître à l'esclave (sans tenir compte de l'expéditeur du message **OpenLogicalChannel**). Pour les canaux médias ouverts par un point d'extrémité (autre que le maître), le maître doit renvoyer la clé de cryptage initiale et le point de synchronisation initial dans le message **OpenLogicalChannelAck** (dans le champ **encryptionSync**). Pour les canaux médias ouverts par le maître, le message **OpenLogicalChannel** doit comporter la clé de cryptage initiale et le point de synchronisation dans le champ **encryptionSync**.

9 Procédures multipoint

9.1 Authentification

L'authentification doit être effectuée entre un point d'extrémité et de pont de conférence de la même façon qu'elle le serait dans une conférence point à point. Le pont doit déterminer la politique concernant le niveau et la sévérité de l'authentification. Comme indiqué au 6.6, le pont est un élément qui doit être crédibilisé. Les points d'extrémité d'une conférence peuvent être limités par le niveau d'authentification employé par le pont de conférence. De nouvelle commandes **ConferenceRequest/ConferenceResponse** permettent aux points d'extrémité d'obtenir du pont les certificats d'autres participants à la conférence. Comme indiqué dans les procédures H.245, les points d'extrémité d'une conférence multipoint peuvent demander d'autres certificats de point d'extrémité via le pont mais ne sont pas toujours en mesure d'effectuer une authentification cryptographique directe à l'intérieur du canal H.245.

9.2 Secret des communications

Un pont de conférence doit remporter tous les échanges maître/esclave et doit donc fournir la ou les clés de cryptage aux participants à une conférence multipoint. Le secret des communications pour des sources individuelles au cours d'une session commune (dans l'hypothèse de destinations multiples) peut être obtenu avec des clés individuelles ou avec des clés communes. Ces deux modes peuvent être arbitrairement choisis par le pont. Ils ne doivent pas pouvoir être commandés à partir d'un point d'extrémité particulier, sauf dans les modes autorisés par la politique des ponts de conférence. En d'autres termes, une clé commune peut être utilisée sur de multiples canaux logiques qui ont été ouverts par des sources différentes.

10 Signalisation et procédures d'authentification

10.1 Introduction

L'authentification est généralement fondée sur une méthode à secret partagé (la connaissance de cette information secrète permet d'être authentifié) ou à clé publique avec certifications (la possession de la clé privée est la preuve de l'identité). Un secret partagé et l'emploi subséquent de la cryptographie symétrique nécessitent un contact préalable entre les entités communicantes. Un face à face préalable ou un contact sécurisé peut être remplacé par la production ou l'échange de la clé d'information secrète par des méthodes fondées sur la cryptographie à clé publique, par exemple l'échange de clés Diffie-Hellman. Pour la production et l'échange de la clé, les parties communicantes doivent être authentifiées, par exemple au moyen de messages à signature numérique; à défaut, les parties communicantes ne savent pas avec certitude avec qui elles partagent l'information secrète.

La présente Recommandation propose des méthodes d'authentification fondées sur l'abonnement, c'est-à-dire qu'il faut un contact préalable pour partager une information secrète et des méthodes d'authentification dans lesquelles la cryptographie par clé publique est utilisée directement pour authentification ou pour la production du secret partagé.

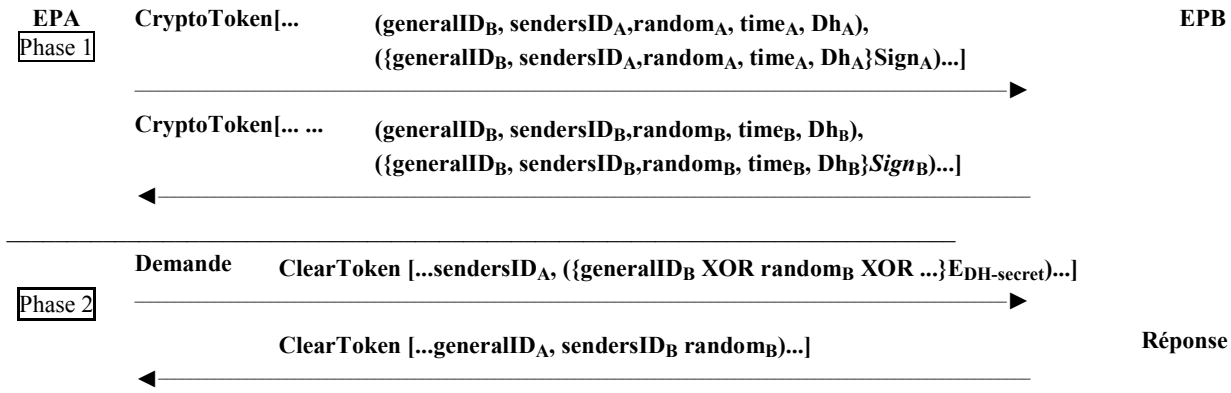
10.2 Méthode de Diffie-Hellman avec authentification facultative

Cette méthode ne vise pas à réaliser une authentification absolue au niveau de l'utilisateur. Elle assure une signalisation permettant de constituer un secret partagé entre deux entités, pouvant conduire à des données de calcul de clés pour des communications privées.

A l'issue de cet échange, les deux entités posséderont une clé à secret partagé ainsi qu'un algorithme sélectionné qui leur permettra d'utiliser cette clé. Cette clé à secret partagé pourra ensuite être utilisée dans tout échange ultérieur de type demande/réponse. Il convient de noter que, dans de rares cas, l'échange Diffie-Hellman peut produire des clés notoirement *faibles* pour certains algorithmes. Dans ces cas, chaque entité devrait se déconnecter et se reconnecter afin d'établir un nouveau jeu de clés.

La première phase de la Figure 1 montre les données échangées lors d'une authentification Diffie-Hellman. La deuxième phase permet au répondant d'authentifier des messages de demande propres à une application ou à un protocole. On notera qu'une nouvelle valeur aléatoire peut être renvoyée avec chaque réponse.

NOTE – Si les messages sont échangés sur un canal non sécurisé, il faut utiliser des signatures numériques (ou toute autre méthode d'authentification de l'origine) pour identifier les parties entre lesquelles l'information secrète sera partagée. Un élément facultatif de signature (indiqué ci-dessous en ***italiques***) peut aussi être fourni.



[... ...] indique une séquence de jetons.

() indique un jeton particulier, qui peut contenir des éléments multiples.

{} **E$_{DH-secret}$** indique que les valeurs contenues sont cryptées au moyen de la méthode de secret Diffie-Hellman.

Le point d'extrémité B (EPB) connaît la clé à secret partagé qu'il faut utiliser pour déchiffrer l'identificateur **generalID$_B$** en l'associant à l'identificateur **generalID$_A$**, qu'il convient de transmettre également dans le message **sendersID$_A$**. On notera que la valeur cryptée dans la phase 2 est transmise dans le champ d'identificateur général **generalID** d'un jeton en clair **clearToken**, afin de simplifier le codage.

Figure 1/H.235 – Echange Diffie-Hellman avec authentification facultative

10.3 Authentification sur abonnement

10.3.1 Introduction

Bien que les procédures décrites ici (ainsi que les algorithmes ISO dont elles sont issues) soient de nature bidirectionnelle, elle ne peuvent être utilisées que dans un seul sens si l'authentification n'est requise que dans ce sens. Les procédures en deux et en trois passages sont décrites. L'authentification mutuelle en deux passages peut être faite dans un sens seulement si les messages provenant du sens opposé ne doivent pas être authentifiés. Ces échanges partent du principe que chaque extrémité possède un certain identificateur notoire (comme un identificateur en mode texte) qui l'identifie sans équivoque. Dans le cas de la procédure en deux passages, une autre hypothèse est faite pour

supposer qu'il existe une référence temporelle acceptable de part et d'autre (permettant de calculer les pointeurs temporels). La grandeur de la dérive temporelle acceptable relève d'une décision de mise en œuvre locale. La procédure en trois passages utilise un numéro d'épreuve imprévisible, produit aléatoirement (pouvant être augmenté d'une "valeur aléatoire" d'un compteur séquentiel), soit une épreuve proposée par l'authentificateur. Ce nombre aléatoire est destiné à la protection contre les agressions par répétition. Contrairement aux procédures en deux passages, les procédures en trois passages n'authentifient pas le premier message (initial) contenant le numéro d'épreuve de l'expéditeur.

Il existe trois variantes différentes de mise en œuvre, selon les exigences:

1) authentification par mot de passe avec cryptage symétrique;

2) authentification par mot de passe avec hachage;

3) authentification par certificat avec signatures.

Dans tous les cas, le jeton contiendra les informations décrites dans les sous-paragraphes suivants, selon la variante choisie. On notera que, dans tous les cas, l'identificateur **generalID** peut être connu par examen de la configuration ou d'un répertoire, plutôt que par échange protocolaire dans la bande. Pour simplifier le traitement au niveau du destinataire, l'expéditeur doit inclure son identité dans l'identificateur **sendersID** et mettre l'identificateur **generalID** à l'identification du destinataire.

NOTE 1 – Chaque fois que des horodateurs sont produits et transmis dans le cadre de l'échange de sécurité, le réalisateur doit prendre les précautions suivantes: la granularité de l'horodateur doit être suffisamment fine pour garantir l'incrémentation à chaque nouveau message. En l'absence de cette garantie, des agressions par répétition sont possibles (par exemple, si l'horodateur n'augmente qu'en minutes, un point d'extrémité "C" peut délibérément tenter de perturber un point d'extrémité "A" pendant la minute qui suit le moment où le point d'extrémité "A" a envoyé un message au point d'extrémité "B").

NOTE 2 – Si le message est multidiffusé, il n'est pas sécurisé.

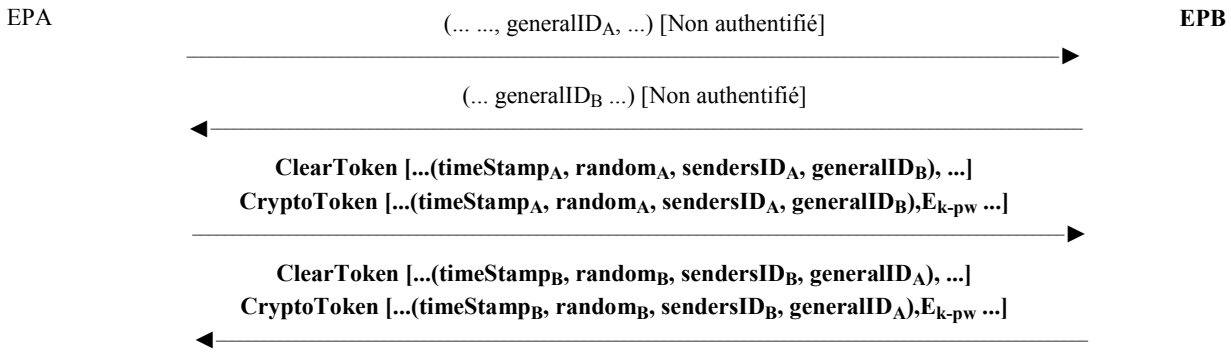
10.3.2 Authentification par mot de passe avec cryptage symétrique

Les Figures 2a et 2b montrent le format du jeton et l'échange de messages requis pour exécuter ce type d'authentification, respectivement en deux et en trois passages. Ce protocole est fondé sur les 5.2.1 (deux passages) et 5.2.2 (trois passages) de l'ISO/CEI 9798-2. On suppose qu'un identificateur et le mot de passe associé sont échangés lors de l'abonnement. La clé de cryptage a une longueur de N octets (comme indiqué par l'identificateur d'algorithme). Elle est formée comme suit:

– si la longueur du mot de passe = N, clé = mot de passe;

– si la longueur du mot de passe < N, la clé est bourrée de zéros;

– si la longueur du mot de passe > N, les N premiers octets sont attribués à la clé, puis le $N + M^e$ octet du mot de passe est combiné par un OU exclusif avec le $M \bmod (N)^e$ octet (pour tous les octets au-delà de N). (En d'autres termes, tous les octets "surnuméraires" du mot de passe sont successivement repliés sur la clé par application de la fonction OUX.)



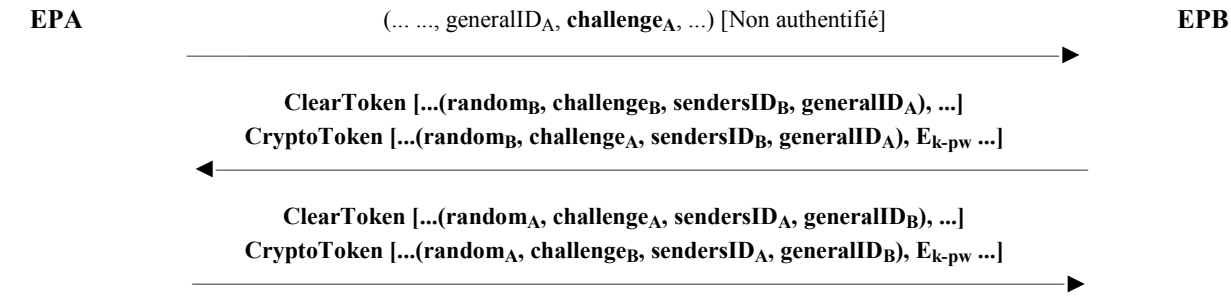
NOTE 1 – Le jeton de retour provenant du point d'extrémité EPB est facultatif; s'il est omis, l'authentification est à sens unique.

NOTE 2 – La variable **E_{k-pw}** indique des valeurs qui sont cryptées au moyen de la clé "k" calculée à partir du mot de passe "pw".

NOTE 3 – **random** est un compteur progressif monotone qui confère l'unicité à plusieurs messages au moyen d'un seul et même horodateur.

NOTE 4 – Dans le troisième message, le point EPA fournit un **ClearToken** distinct qui est identifié au moyen du même identificateur OID que celui de **CryptoToken**; il en est de même pour le quatrième message et inversement.

Figure 2a/H.235 – Authentification par mot de passe avec cryptage symétrique – Deux passages



NOTE 1 – L'épreuve **challenge$_A$** et le **CryptoToken** crypté envoyé en retour par B à A ne sont pas nécessaires en cas d'authentification à sens unique.

NOTE 2 – La variable **E_{k-pw}** indique une fonction de cryptage qui est cryptée au moyen de la clé "k" calculée à partir du mot de passe "pw".

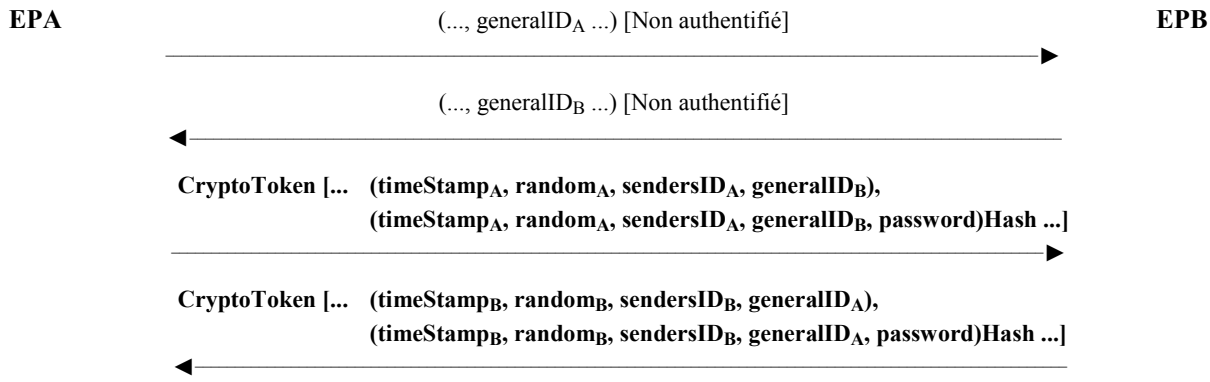
NOTE 3 – Dans le troisième message, le point EPA envoie une nouvelle épreuve **challenge$_A$** en clair dans un **ClearToken** distinct qui est identifié au moyen du même identificateur OID que celui de **CryptoToken**. Le point EPA renvoie également, en réponse, l'épreuve **challenge$_B$** cryptée; il en est de même pour le deuxième message et inversement.

NOTE 4 – S'il y a plusieurs messages en attente, l'épreuve doit être rendue unique par **random** (c'est-à-dire un compteur progressif monotone)

Figure 2b/H.235 – Authentification par mot de passe avec cryptage symétrique – Trois passages

10.3.3 Authentification par mot de passe avec hachage

Les Figures 3a et 3b montrent le format du jeton et l'échange du message requis pour exécuter ce type d'authentification, respectivement en deux et en trois passages. Ce protocole est fondé sur les § 5.2.1 et 5.2.2 de l'ISO/CEI 9798-4; on suppose qu'un identificateur et le mot de passe associé sont échangés au moment de l'abonnement. L'Annexe D contient la description détaillée de la procédure de hachage en deux passages.

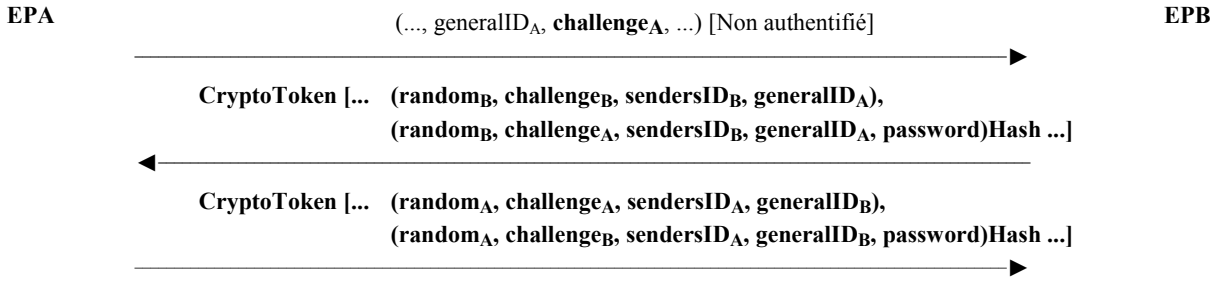


NOTE 1 – Le jeton de retour de l'extrémité EPB est facultatif; s'il est omis, l'authentification est à sens unique.

NOTE 2 – La variable **Hash** indique une fonction de hachage qui agit sur les valeurs contenues.

NOTE 3 – **random** est un compteur progressif monotone qui confère l'unicité à plusieurs messages avec un seul et même horodateur.

Figure 3a/H.235 – Authentification par mot de passe avec hachage – Deux passages



NOTE 1 – Le jeton de retour de l'extrémité EPB est facultatif; s'il est omis, l'authentification est à sens unique.

NOTE 2 – La variable **Hash** indique une fonction de hachage qui agit sur les valeurs contenues.

NOTE 3 – Dans le troisième message, le point EPA envoie une nouvelle épreuve **challenge$_A$** en clair dans le **ClearToken** intégré dans **cryptoHashedToken**. Le point EPA renvoie l'épreuve **challenge$_B$** hachée à titre de réponse; il en est de même pour le deuxième message et inversement.

NOTE 4 – S'il y a plusieurs messages en attente, l'épreuve doit être rendue unique par **random** (compteur progressif monotone).

Figure 3b/H.235 – Mot de passe avec hachage – Trois passages

NOTE 1 – La structure **cryptoHashedToken** est utilisée pour le transfert des paramètres utilisés dans cet échange. Les versions "en clair" des paramètres nécessaires pour calculer la valeur hachée sont incluses dans cette structure. Les réalisateurs doivent inclure l'horodateur dans les **hashedVals** et *ne* doivent *pas* inclure le mot de passe (par exemple, le mot de passe et le "**generalID**" devraient être connus *a priori* par le destinataire; ce qui précède peut être omis).

NOTE 2 – La fonction de hachage doit être appliquée à la structure **EncodedGeneralToken** qui englobe au moins les champs ID, horodateur et mot de passe. La valeur du mot de passe NE doit PAS être acheminée dans **ClearToken**.

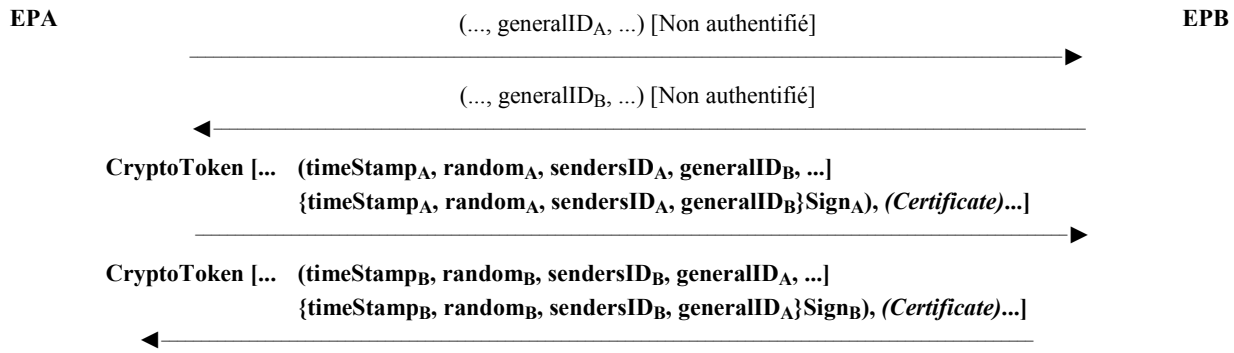
NOTE 3 – Les réalisateurs devraient s'assurer que les mots de passe entrés par l'utilisateur présentent une entropie suffisante. Les mots de passe trop courts ou qui sont sensibles aux attaques de dictionnaires devraient être refusés. Il peut être intéressant, dans certains cas, de faire passer le mot de passe/phrase entré par l'utilisateur par une fonction de hachage cryptographique et d'utiliser les bits de sortie.

10.3.4 Authentification par certificat avec signatures

Les Figures 4a et 4b montrent le format du jeton et l'échange de messages requis pour exécuter ce type d'authentification. Ce protocole est fondé sur le 5.2.1 de l'ISO/CEI 9798-3; on suppose qu'un identificateur et le mot de passe associé sont échangés lors de l'abonnement. L'Annexe E contient la description détaillée de la procédure de signature en deux passages.

NOTE 1 – Un élément facultatif de certificat (indiqué ci-dessous en ***italique***) peut aussi être fourni.

NOTE 2 – Si le message est multidiffusé, l'identificateur de la destination (**generalID$_B$** pour les messages provenant de A et inversement) ne doit pas être inclus dans le jeton **ClearToken**.



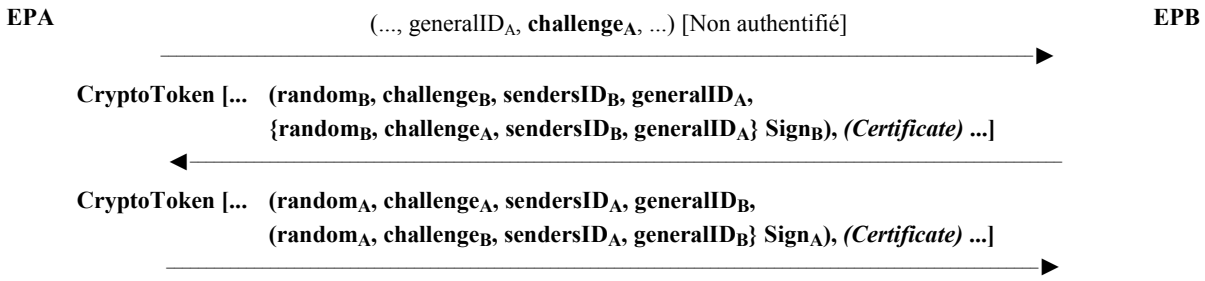
NOTE 1 – Le jeton de retour de l'extrémité EPB est facultatif; s'il est omis, l'authentification est à sens unique.

NOTE 2 – Un certificat de type "payment" peut, facultativement, être inclus par l'expéditeur situé au point EPA.

NOTE 3 – La variable **Sign** indique une fonction de signature (issue du certificat associé) qui est exécutée sur les valeurs contenues.

NOTE 4 – **random** est un compteur progressif monotone qui confère l'unicité à plusieurs messages au moyen d'un seul et même horodateur.

Figure 4a/H.235 – Authentification par certificat avec signature – Deux passages



NOTE 1 – Le jeton de retour de l'extrémité EPB est facultatif; s'il est omis, l'authentification est à sens unique.

NOTE 2 – Un certificat de type "payment" peut, facultativement, être inclus par l'expéditeur situé au point EPA.

NOTE 3 – La variable **Sign** indique une fonction de signature (issue du certificat associé) qui est exécutée sur les valeurs contenues..

NOTE 4 – Dans le troisième message, le point EPA envoie une nouvelle épreuve **challenge$_A$** en clair avec le **GeneralToken** codé incorporé. Le point EPA renvoie également l'épreuve **challenge$_B$** signée en réponse; il en est de même pour le deuxième message et inversement.

NOTE 5 – Si plusieurs messages sont en attente, l'épreuve doit être rendue unique par **random** (compteur progressif monotone).

Figure 4b/H.235 – Authentification par certificat avec signature – Trois passages

10.3.5 Utilisation du secret partagé et des mots de passe

Dans la présente Recommandation, certaines techniques cryptographiques symétriques sont appliquées aux fins de l'authentification, de l'intégrité et de la confidentialité. Les termes "mot de passe" et "secret partagé" sont ici employés lorsqu'on utilise des techniques symétriques. On entend par le terme générique de "secret partagé" une chaîne binaire arbitraire. Cette chaîne peut être attribuée ou configurée au moment de la souscription de l'utilisateur ou faire partie d'un calcul d'authentification, par exemple de type Diffie-Hellman.

Un mot de passe pourrait être assimilé à une chaîne de caractères alphanumériques qui peut être mémorisée par les utilisateurs. Il va sans dire que l'utilisation des mots de passe ne va pas sans certaines précautions: pour offrir des garanties de sécurité suffisantes de telle sorte qu'ils ne puissent pas être découverts, et enfin être régulièrement modifiés. Les règles de création et de mise à jour des mots de passe n'entrent pas dans le cadre de la présente Recommandation.

Une méthode efficace pour tirer parti des mots de passe et des secrets partagés consiste à transformer la chaîne du mot de passe de l'utilisateur en une chaîne binaire fixe qui devient ainsi le secret partagé, au moyen d'une fonction de hachage unilatéral robuste sur le plan cryptographique.

A titre d'exemple, dans le cas du profil de sécurité visé dans l'Annexe D, la fonction de hachage SHA-1 appliquée à la chaîne du mot de passe produit un secret partagé à 20 octets. Le hachage présente l'avantage de non seulement occulter le mot de passe proprement dit mais aussi de définir un format de chaîne binaire à longueur fixe sans réellement supprimer l'entropie pour autant.

Par conséquent,

secret partagé: = SHA1 (mot de passe)

11 Procédures de cryptage de flux médias

Les flux médias doivent être codés au moyen de l'algorithme et de la clé qui sont présentés dans le canal H.245. Les Figures 5 et 6 montrent le flux général. On notera que l'en-tête de transport est attaché à l'unité SDU de transport une fois que cette unité a été cryptée. Les segments opaques indiquent le secret des communications. Au fur et à mesure que de nouvelles clés sont reçues par l'émetteur et utilisées dans le cryptage, l'en-tête d'unité SDU doit indiquer d'une façon ou d'une autre au récepteur que la nouvelle clé est désormais en usage. Par exemple, dans un flux UIT-T H.323, l'en-tête (SDU) de protocole RTP modifiera son type de capacité utile pour indiquer le commutateur à la nouvelle clé.

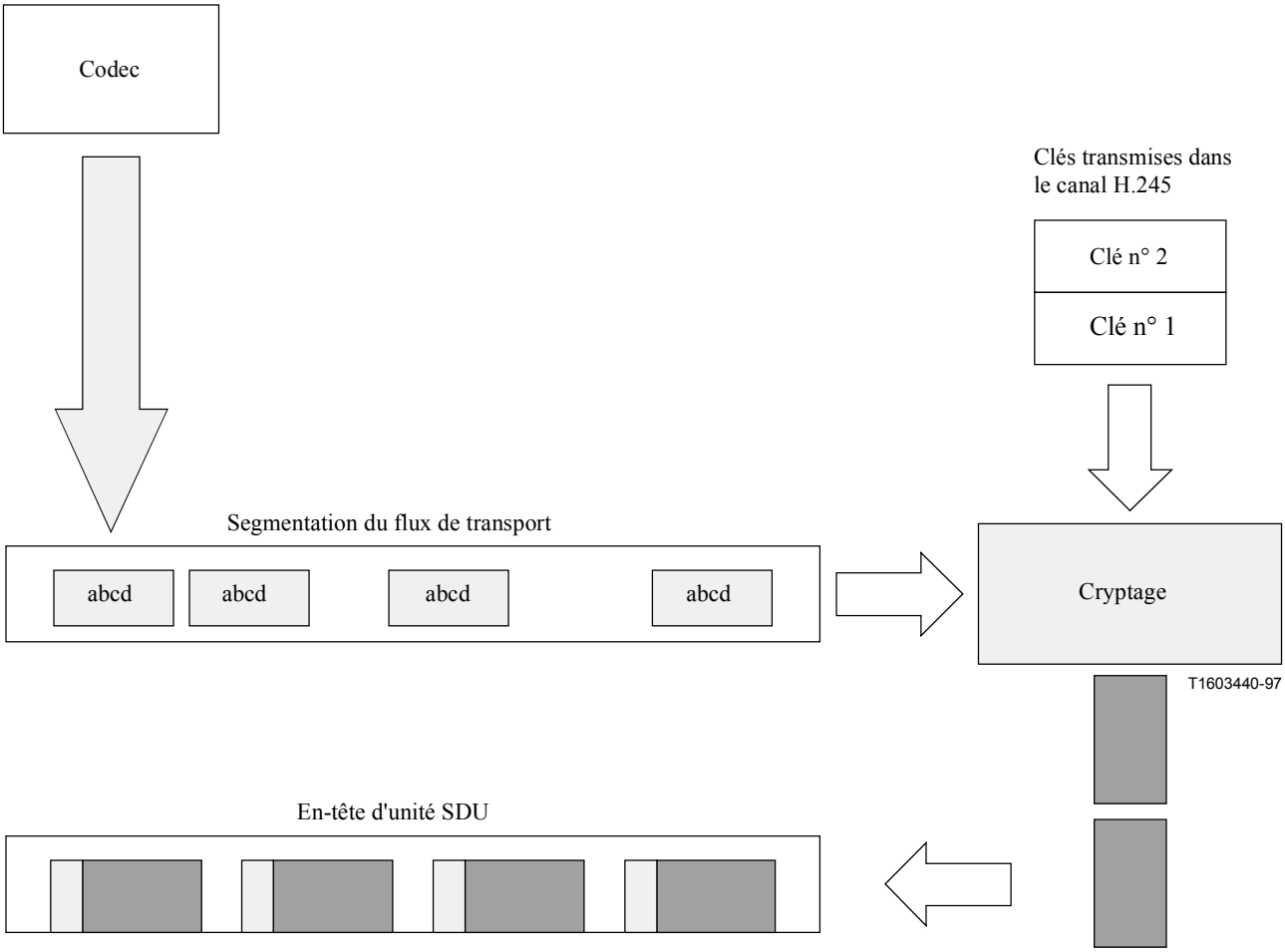


Figure 5/H.235 – Cryptage du média

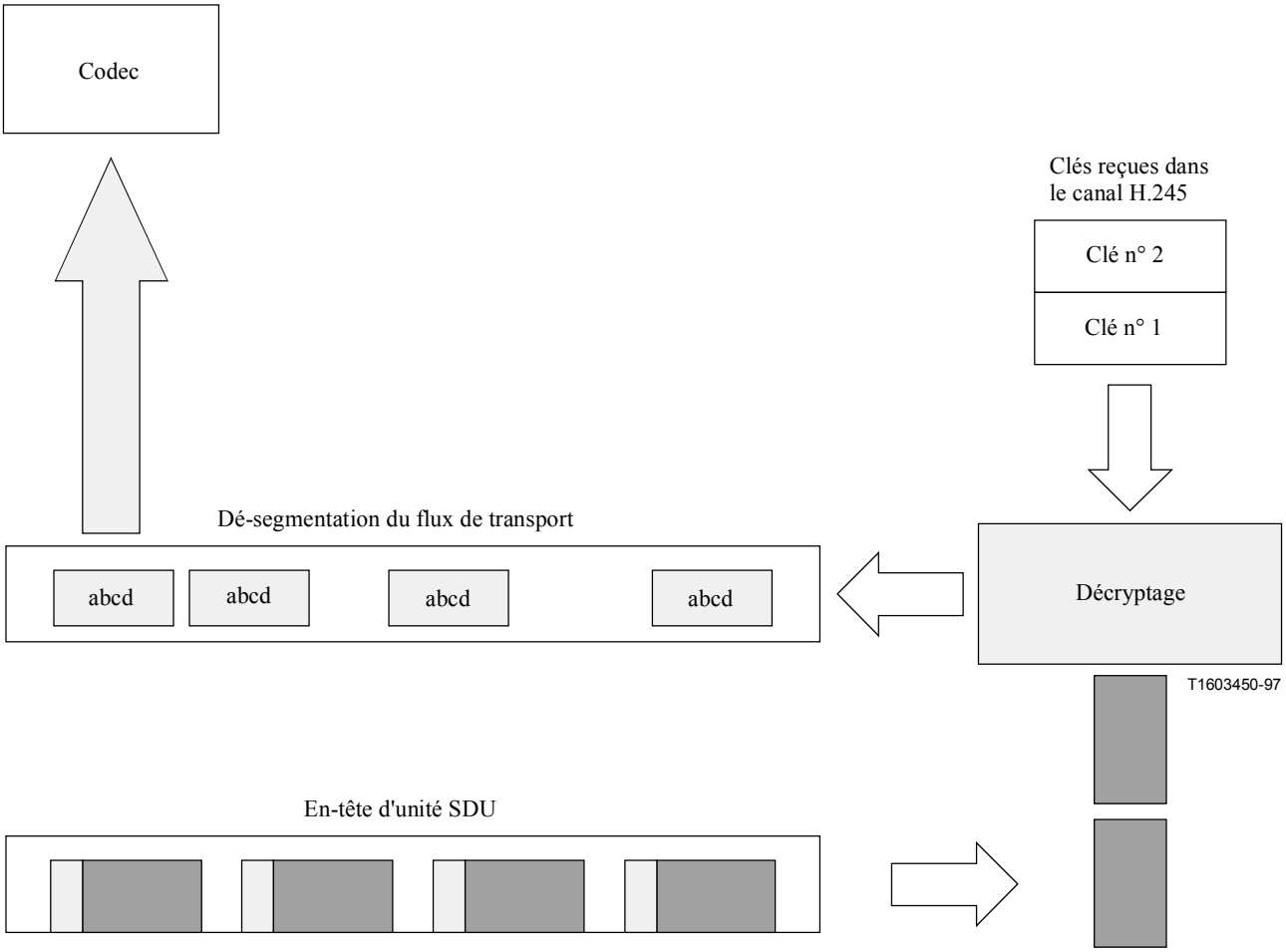


Figure 6/H.235 – Décryptage du média

11.1 Clés de session média

Le message **encryptionUpdate** comporte le champ de clé **h235key**, qui est codé en notation ASN.1 dans le contexte de l'arbre ASN.1 de l'UIT-T H.235 et qui est transmis sous forme d'une chaîne d'octets opaque par rapport au flux H.245. Cette clé peut être protégée au moyen d'un des trois mécanismes possibles, au fur et à mesure de leur passage entre deux points d'extrémité.

- si le canal H.245 est sécurisé, aucune protection additionnelle n'est appliquée aux données de clé. Celle-ci est transmise "en clair" dans ce champ; la valeur de choix ASN.1 **secureChannel** est alors utilisée;
- si une clé et un algorithme de secret ont été établis en dehors du canal H.245 dans son ensemble (c'est-à-dire hors du flux H.323 ou sur un canal logique de type **h235Control**), le secret partagé est utilisé pour chiffrer les données de clé et la clé chiffrée résultante est insérée dans ce champ. Dans ce cas, la valeur de choix ASN.1 **sharedSecret** est utilisée;
- des certificats peuvent être utilisés lorsque le canal H.245 n'est pas sécurisé; mais ils peuvent aussi être employés en complément d'un canal H.245 sécurisé. Lorsque des certificats sont utilisés, les données de clé sont chiffrées au moyen de la clé publique du certificat et de la structure ASN.1 **certProtectedKey**.

A tout point d'une conférence, un récepteur (ou un émetteur) peut demander une nouvelle clé (par une demande de type **encryptionUpdateRequest**), par exemple parce qu'il suppose qu'il a perdu la synchronisation de l'un des canaux logiques. Le point maître qui reçoit cette demande doit produire une ou des nouvelles clés en réponse à cette commande. Le maître peut également décider, de manière asynchrone, de distribuer une ou des nouvelles clés: il doit dans ce cas utiliser le message **encryptionUpdate**.

Après avoir reçu une demande **encryptionUpdateRequest**, un maître doit envoyer une mise à jour **encryptionUpdate**. Si la conférence est de type multipoint, le pont (en tant que maître) doit normalement distribuer la nouvelle clé à tous les récepteurs avant de la donner à l'émetteur. L'émetteur des données sur le canal logique doit utiliser la nouvelle clé au plus tôt après avoir reçu le message.

Un émetteur (supposé autre que le maître) peut également demander une nouvelle clé. Si l'émetteur fait partie d'une conférence multipoint, la procédure doit être la suivante:

- l'émetteur doit envoyer au pont (maître) la demande **encryptionUpdateRequest;**
- le pont doit produire une ou des nouvelles clés et envoyer un message **encryptionUpdate** à tous les participants de la conférence, sauf à l'émetteur;
- après avoir distribué les nouvelles clés à tous les autres participants, le pont doit envoyer le message **encryptionUpdate** à l'émetteur. Celui-ci doit alors utiliser la nouvelle clé.

11.2 Protection du média contre la submersion

Le destinataire d'un flux de média RTP voudra peut-être s'opposer à des agressions visant la fonction de refus de service et des agressions par submersion constatées au niveau des accès RTP/UDP. Lorsqu'il a implémenté la capacité anti-submersion, le destinataire peut rapidement déterminer si un paquet RTP reçu provient d'une source non autorisée et le refuser.

Lorsqu'elle est active, la capacité anti-submersion signale que le mécanisme correspondant est utilisé:

- pour des données en clair sans cryptage (voir le cas 1 ci-dessous);
- en combinaison avec des données cryptées lorsque **EncryptionCapability** contient un algorithme de cryptage (voir le cas 2 ci-dessous).

Les deux possibilités offrent une authentification **RTP packet authentication** modérée de champs sélectionnés au moyen d'un code d'identification de messages calculés (MAC, *message authentication code*). Ce code peut être calculé au moyen des identificateurs d'objets définis au 11.2.1. Le cryptage est obtenu:

- par un algorithme de cryptage (tel que DES en mode MAC; voir ISO/CEI 9797). Le code DES-MAC est signalé au moyen de l'indicateur OID "S", alors que le code DES-MAC triple est signalé au moyen de l'indicateur OID "0";
- par l'emploi d'une fonction cryptographique directionnelle (telle que SHA1). L'indicateur OID qu'il convient d'utiliser est "M".

L'algorithme MAC est indiqué dans l'identificateur d'objet de l'algorithme **antiSpamAlgorithm**. L'identificateur OID de l'algorithme indique en outre, implicitement, la taille du code MAC: par exemple, 1 bloc = 64 bits pour le code DES-MAC. Pour économiser de la largeur de bande, le code MAC peut être tronqué moyennant une légère diminution de la sécurité, de manière à former un code MAC à 32 bits par exemple; cela nécessite un identificateur d'objet différent. La méthode anti-submersion est indépendante de tout cryptage additionnel de la capacité utile (voir les cas 1 et 2 ci-dessous).

La protection contre la submersion utilise le format de paquets RTP ci-après (voir Figure 7) lorsque la séquence de remplissage du RTP est interprétée de la manière suivante (voir Annexe A.5/H.225.0).

- Le bit P de l'en-tête du RTP doit être mis à 1.
- Les octets de remplissage doivent être ajoutés à la fin de la capacité utile avec la signification suivante:

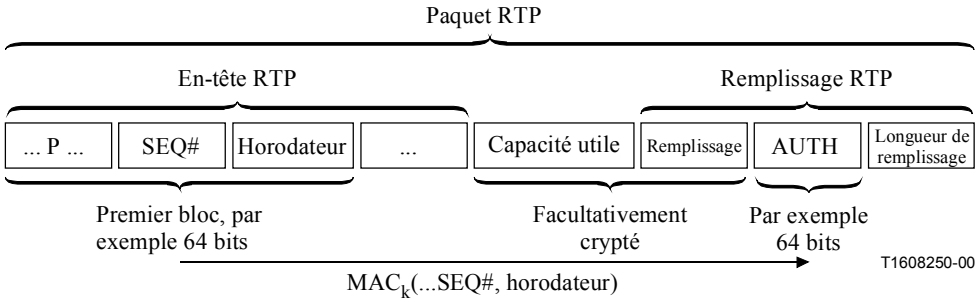


Figure 7/H.235 – Format de paquets RTP pour la protection contre la submersion du média

NOTE 1 – Lorsque la protection contre la submersion n'est pas utilisée, les champs "AUTH" et "*padlen*" ne sont pas utilisés eux non plus et le format de paquets RTP usuel s'applique.

1) Cas de la protection anti-submersion utilisée seule:

ce cas s'applique lorsque les données médias ne sont pas cryptées et que les champs de remplissage restent vides. Le dernier octet de remplissage RTP contient le décompte des octets de remplissage qu'il convient d'ignorer à la fin du paquet RTP. Les autres octets de remplissage acheminent le code MAC. Celui-ci doit être calculé sur le premier bloc crypté de l'en-tête RTP comprenant l'horodateur et le numéro de séquence variable utilisant l'algorithme MAC négocié de **antiSpamAlgorithm** et appliquant le secret symétrique. Un secret partagé statique ou configuré manuellement ou un secret partagé *k* négocié dynamiquement peut être utilisé conformément aux procédures H.235. Pour des tailles de bloc plus importantes (supérieures à 64 bits), il faudra prendre un nombre de bits additionnels suffisant de l'en-tête RTP, voire de la première capacité utile.

Pour le calcul du code MAC, il est recommandé d'utiliser la clé obtenue lors de la distribution de clés de la session média H.235, bien que la clé de session appliquée ne soit pas utilisée pour le cryptage de la capacité utile. On peut utiliser, pour la gestion des clés, la connexion rapide sécurisée avec établissement des clés (voir l'Annexe J/H.323) ou le mode manuel. L'expéditeur calcule le code MAC comme indiqué ci-dessus et inclut le résultat dans le champ MAC du champ AUTH de remplissage du RTP. L'expéditeur et le destinataire connaissent la taille du champ AUTH et la longueur du code MAC par **antiSpamAlgorithm**.

La vérification du code MAC du côté destinataire doit être faite le plus tôt possible, éventuellement dans la pile RTP ou, au plus tard, avant le cryptage ou la décompression de la capacité utile. Le destinataire recalcule d'abord le code MAC de la même manière que l'a fait l'expéditeur et compare le code MAC calculé avec le code MAC remis dans le remplissage RTP. Si les codes MAC ne concordent pas, l'en-tête de paquet RTP a été modifié pendant le transport ou a été envoyé par une entité non autorisée qui ne possède pas la clé. Donc, les paquets RTP ne pouvant être authentifiés doivent être ignorés et l'événement peut être consigné; cela indique probablement une tentative d'agression de la

fonction de refus de service. Sinon, le traitement du paquet RTP authentifié peut se poursuivre, le remplissage RTP est supprimé et la capacité utile est envoyée dans le codec.

NOTE 2 – Le calcul/vérification sommaire du code MAC avec cryptage DES fait intervenir une seule opération de cryptage; à l'inverse, le codage SHA1 MAC est calculé sur une partie réduite des paquets de longueur fixe; les opérations de cryptage consomment donc un minimum de ressources de traitement.

2) Cas de la méthode anti-submersion avec cryptage de la capacité utile:

ce cas s'applique lorsque les données médias sont cryptées et que la méthode anti-submersion est sollicitée. Lorsque la capacité utile ne correspond pas à des limites de blocs paires, certains octets de remplissage additionnels doivent être ajoutés à la capacité utile, devant le code MAC. Le cryptage de la capacité utile média est conforme au paragraphe 11.

EncryptionCapability définit l'algorithme de cryptage de la capacité utile alors que **antiSpamAlgorithm** définit la méthode anti-submersion. Pour des raisons de sécurité, le cryptage du média et le code MAC doivent utiliser des clés de session différentes. La clé *k* du code MAC est calculée en introduisant la clé de cryptage *K* dans la fonction de hachage unidirectionnelle unique SHA1.

k = SHA1(*K*); il convient de prendre suffisamment de bits sur le résultat du hachage, dans l'ordre des octets du réseau. Lorsque **antiSpamAlgorithm** indique un algorithme de cryptage, les bits collectés doivent être transformés en clé de cryptage correcte; par exemple, en posant les bits de parité de la norme DES.

Lorsque le destinataire a correctement vérifié l'authenticité du paquet RTP, la capacité utile est décryptée et le remplissage RTP est écarté. La procédure générale est conforme au cas 1 ci-dessus.

11.2.1 Liste des identificateurs d'objet

Le Tableau 1 énumère tous les identificateurs d'objet OID qui ont été cités:

Tableau 1/H.235 – Identificateurs d'objet utilisés pour la protection contre la submersion

Référence de l'identificateur d'objet	Valeur de l'identificateur d'objet	Description
"M"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 8}	anti-submersion utilisant le code HMAC-SHA1-96
"N"	{iso(1), identified-organization(3), oiw(14), secsig(3), algorithm(2), desMAC(10)}	anti- submersion utilisant le code MAC DES (56 bits) (voir ISO/CEI 9797) avec code MAC à 64 bits.
"O"	{iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) desEDE(17)}	anti- submersion utilisant le code MAC-DES triple (168 bits) (voir ISO/CEI 9797)

12 Reprise sur erreur de sécurité

La présente Recommandation ne spécifie ni ne préconise de méthodes permettant aux points d'extrémité de surveiller le secret absolu de leurs communications. Elle recommande cependant des mesures à prendre lors de la détection d'une perte du secret des communications.

Si l'un des points d'extrémité détecte une brèche dans la sécurité du canal de connexion d'appel (par exemple un canal H.225.0 pour flux H.323), il doit immédiatement fermer la connexion conformément aux procédures protocolaires appropriées au point d'extrémité en question [pour un flux, selon 8.5/H.323, à l'exception du bond 5)].

Si l'un des points d'extrémité détecte une brèche dans la sécurité du canal H.245 ou du canal logique à données sécurisées (**h235Control**), il doit immédiatement fermer la connexion conformément aux procédures protocolaires appropriées au point d'extrémité en question [pour un flux H.323, selon le 8.5/H.323 à l'exception du bond 5)].

Si l'un des points d'extrémité détecte une perte du secret des communications sur l'un des canaux logiques, il doit immédiatement demander une nouvelle clé (par une demande **encryptionUpdateRequest**) et/ou fermer le canal logique. A la discrétion du pont de conférence, une perte de secret sur un canal logique peut causer la fermeture de tous les autres canaux logiques et/ou le recalcul de leurs clés. Le pont de conférence doit envoyer une demande de mise à jour **encryptionUpdateRequest** et une mise à jour **encryptionUpdate** à tous les points d'extrémités affectés.

A la discrétion du pont de conférence, une erreur de sécurité sur un canal individuel peut provoquer la fermeture des connexions à tous les points d'extrémité de la conférence – ce qui met fin à celle-ci.

13 Authentification asymétrique et échange de clés au moyen de systèmes de cryptage à courbe elliptique

La présente Recommandation propose des techniques elliptiques élaborées s'appliquant à la signature, à la gestion des clés et au cryptage. Un des principaux avantages par rapport aux techniques asymétriques "classiques" telles que l'algorithme RSA sont:

- des clés cryptographiques plus courtes assurant une sécurité comparable à celle de l'algorithme RSA: généralement, la longueur des clés des systèmes à courbe elliptique est de 160 bits, soit l'équivalent, au plan de la sécurité, à une clé RSA de 1024 bits. La clé plus courte consomme moins de mémoire de stockage et rend l'utilisation des systèmes cryptographiques à courbe elliptique particulièrement attrayants dans les cartes à puce et autres dispositifs à faible besoin de mémoire. Dans le contexte H.323, les dispositifs d'extrémité simples sonores sécurisés (SASET, *secured audio simple endpoint type*) sont de type Annexe J/H.323; peu onéreux, ils conviennent fort bien au déploiement des techniques à courbe elliptique.
- La grande vitesse de traitement atteinte tant au niveau du logiciel que du matériel: les clés plus courtes favorisent la vitesse de traitement, ce qui se traduit par une réponse interactive (utilisateur) plus rapide.

Tous les renseignements généraux, explications et procédures de traitement de la cryptographie elliptique sont donnés dans [*ATM Forum Security Specification Version 1.1,* section 8.7]. Il est recommandé de coder les points elliptiques dans leur notation affine, non comprimée, sans la méthode du point de compression/décompression. D'autres informations à ce sujet figurent dans [ISO/CEI 15946-1] et [ISO/CEI 15946-2].

13.1 Gestion de clés

Les systèmes de concordance de clés de type Diffie-Hellman elliptiques sont analogues au cas classique mod-p, également défini dans la présente Recommandation. Deux cas peuvent se présenter:

- courbes elliptiques sur un champ principal: **eckasdhp** contient la courbe elliptique et les paramètres Diffie-Hellman;

- courbes elliptiques de caractéristique 2: **eckasdh2** contient la courbe elliptique et les paramètres Diffie-Hellman.

La structure ECKASDH se rapporte aux deux cas. Quelques exemples de courbes elliptiques sont énumérés dans [ISO/CEI 15946-1]. Toute autre courbe elliptique appropriée peut être utilisée.

En raison de la structure ordonnée qu'offre le jeton **ClearToken**, la signalisation de **dhkey** et **eckasdhkey** ne doit pas avoir lieu en même temps; un des deux seulement sera présent lors de l'application de l'échange de clés Diffie-Hellman.

Remarque: il ne faut pas confondre les paramètres secrets choisis aléatoirement, ***a*** par la partie A et ***b*** par la partie B, avec les coefficients ***a*** et ***b*** de Weierstrass.

13.2 Signature numérique

Le champ **ECGDSASignature** contient les valeurs **r** et **s** de la signature numérique de type elliptique calculée. La section 8.7.3 de l'*ATM Security Specification Version 1.1* et le chapitre 5 de l'ISO 15946-2 contiennent d'autres informations sur l'EC-GDSA de l'algorithme de signature.

La signature numérique **ECGDSA** de type elliptique doit être codée ASN.1 puis placée dans le champ **signature** de la macro **SIGNED** de la présente Recommandation. En ce qui concerne la signature numérique, l'expéditeur doit inclure l'identificateur d'objet dans **algorithmOID** par lequel le destinataire est en mesure de déterminer l'utilisation d'une signature numérique elliptique.

ANNEXE A

ASN.1 H.235

```
H235-SECURITY-MESSAGES DEFINITIONS AUTOMATIC TAGS ::=
BEGIN

-- EXPORTS All

ChallengeString         ::= OCTET STRING (SIZE(8..128))
TimeStamp               ::= INTEGER(1..4294967295) -- seconds since 00:00 1/1/1970 UTC
RandomVal               ::= INTEGER -- 32-bit Integer
Password                ::= BMPString (SIZE (1..128))
Identifier              ::= BMPString (SIZE (1..128))
KeyMaterial             ::= BIT STRING(SIZE(1..2048))

NonStandardParameter ::= SEQUENCE
{
        nonStandardIdentifier   OBJECT IDENTIFIER,
        data                    OCTET STRING
}

-- if local octet representations of these bit strings are used they shall
-- utilize standard Network Octet ordering (e.g. Big Endian)
DHset ::= SEQUENCE
{
        halfkey         BIT STRING (SIZE(0..2048)), -- = g^x mod n
        modSize         BIT STRING (SIZE(0..2048)), --  n
        generator       BIT STRING (SIZE(0..2048)), -- g
        ...
}

ECpoint ::= SEQUENCE -- uncompressed (x, y) affine coordinate representation of an elliptic curve point
{
        x               BIT STRING (SIZE(0..511)) OPTIONAL,
        y               BIT STRING (SIZE(0..511)) OPTIONAL,
```

```
        ...
}

ECKASDH::= CHOICE -- parameters for elliptic curve key agreement scheme Diffie-Hellman
{
        eckasdhp SEQUENCE -- parameters for elliptic curves of prime field
        {
                public-key          ECpoint, -- This field contains representation of the ECKAS-DHp public key value.
                        --This field contains the initiator's ECKAS-DHp public key value (aP) when this information
                        -- element is sent from originator to receiver. This field contains the responder's ECKAS-DHp
                        -- public key value (bP) when this information element is sent back from receiver
                        -- to originator.
                modulus             BIT STRING (SIZE(0..511)), -- This field contains representation of the
                        -- ECKAS-DHp public modulus value (p).
                base                ECpoint, -- This field contains representation of the ECKAS-DHp public base (P).
                weierstrassA        BIT STRING (SIZE(0..511)), --This field contains representation of the
                        -- ECKAS-DHp Weierstrass coefficient (a).
                weierstrassB        BIT STRING (SIZE(0..511)) --This field contains representation of the
                        -- ECKAS-DHp Weierstrass coefficient (b).
        },

        eckasdh2 SEQUENCE -- parameters for elliptic curves of characteristic 2
        {
                public-key          ECpoint, -- This field contains representation of the ECKAS-DH2 public key value.
                        -- This field contains the initiator's ECKAS-DH2 public key value (aP) when this information
                        -- element is sent from originator to receiver. This field contains the responder's ECKAS-DH2
                        -- public key value (bP) when this information element is sent back from receiver to originator.
                fieldSize           BIT STRING (SIZE(0..511)), -- This field contains representation of the
                        -- ECKAS-DH2 field size value (m).
                base                ECpoint, -- This field contains representation of  the ECKAS-DH2 public base (P).
                weierstrassA        BIT STRING (SIZE(0..511)), --This field contains representation of the
                        -- ECKAS-DH2 Weierstrass coefficient (a).
                weierstrassB        BIT STRING (SIZE(0..511)) --This field contains representation of the
                        -- ECKAS-DH2 Weierstrass coefficient (b).
        },
        ...
}

ECGDSASignature::= SEQUENCE -- parameters for elliptic curve digital signature algorithm
{
        r           BIT STRING (SIZE(0..511)), -- This field contains the representation of the r component of the
                        -- ECGDSA  digital signature.
        s           BIT STRING (SIZE(0..511)) -- This field contains the representation of the s component of the
                        -- ECGDSA  digital signature.
}

TypedCertificate ::= SEQUENCE
{
        type                OBJECT IDENTIFIER,
        certificate         OCTET STRING,
        ...
}

AuthenticationBES ::= CHOICE

{
        default     NULL, -- encrypted ClearToken
        radius      NULL, -- RADIUS-challenge/response
        ...

}
```

```
AuthenticationMechanism ::= CHOICE
{
        dhExch              NULL, -- Diffie-Hellman
        pwdSymEnc           NULL, -- password with symmetric encryption
        pwdHash             NULL, -- password with hashing
        certSign            NULL, -- Certificate with signature
        ipsec               NULL, -- IPSEC based connection
        tls                 NULL,
        nonStandard         NonStandardParameter, -- something else.
        ...,
        authenticationBES   AuthenticationBES -- user authentication for BES
}

ClearToken          ::= SEQUENCE  -- a "token" may contain multiple value types.
{
        tokenOID            OBJECT IDENTIFIER,
        timeStamp           TimeStamp OPTIONAL,
        password            Password OPTIONAL,
        dhkey               DHset OPTIONAL,
        challenge           ChallengeString OPTIONAL,
        random              RandomVal OPTIONAL,
        certificate         TypedCertificate OPTIONAL,
        generalID           Identifier OPTIONAL,
        nonStandard         NonStandardParameter OPTIONAL,
        ...,
        eckasdhkey          ECKASDH OPTIONAL, -- elliptic curve Key Agreement Scheme-Diffie
                                              -- Hellman Analogue (ECKAS-DH)
        sendersID           Identifier OPTIONAL
}

--      An object identifier should be placed in the tokenOID field when a
--      ClearToken is included directly in a message (as opposed to being
--      encrypted).  In all other cases, an application should use the
--      object identifier { 0 0 } to indicate that the tokenOID value is not present.
--
-- Start all the cryptographic parameterized types here...
--

SIGNED { ToBeSigned } ::= SEQUENCE {
        toBeSigned          ToBeSigned,
        algorithmOID        OBJECT IDENTIFIER,
        paramS              Params,       -- any "runtime" parameters
        signature           BIT STRING -- could be an RSA or an ASN.1 coded ECGDSASignature
} ( CONSTRAINED BY { -- Verify or Sign Certificate -- } )


ENCRYPTED { ToBeEncrypted } ::= SEQUENCE {
        algorithmOID            OBJECT IDENTIFIER,
        paramS                  Params,       -- any "runtime" parameters
        encryptedData           OCTET STRING
} ( CONSTRAINED BY { -- Encrypt or Decrypt -- ToBeEncrypted } )

HASHED { ToBeHashed } ::= SEQUENCE {
        algorithmOID            OBJECT IDENTIFIER,
        paramS                  Params,       -- any "runtime" parameters
        hash                    BIT STRING
} ( CONSTRAINED BY { -- Hash -- ToBeHashed } )

IV8 ::= OCTET STRING (SIZE(8)) -- initial value for 64-bit block ciphers
IV16 ::= OCTET STRING (SIZE(16)) -- initial value for 128-bit block ciphers

-- signing algorithm used must select one of these types of parameters
-- needed by receiving end of signature.
```

```
Params ::= SEQUENCE {
        ranInt              INTEGER OPTIONAL, -- some integer value
        iv8                 IV8 OPTIONAL,    -- 8 octet initialization vector
        ...,
        iv16                IV16 OPTIONAL   -- 16 octet initialization vector
}

EncodedGeneralToken ::= TYPE-IDENTIFIER.&Type (ClearToken -- general usage token -- )
PwdCertToken ::= ClearToken (WITH COMPONENTS {..., timeStamp PRESENT, generalID PRESENT})
EncodedPwdCertToken ::= TYPE-IDENTIFIER.&Type (PwdCertToken)

CryptoToken::= CHOICE
{

        cryptoEncryptedToken SEQUENCE -- General purpose/application specific token
        {
                tokenOID    OBJECT IDENTIFIER,
                token       ENCRYPTED { EncodedGeneralToken }
        },
        cryptoSignedToken  SEQUENCE -- General purpose/application specific token
        {
                tokenOID    OBJECT IDENTIFIER,
                token       SIGNED { EncodedGeneralToken }
        },
        cryptoHashedToken SEQUENCE -- General purpose/application specific token
        {
                tokenOID            OBJECT IDENTIFIER,
                hashedVals          ClearToken,
                token HASHED { EncodedGeneralToken }
        },
        cryptoPwdEncr       ENCRYPTED { EncodedPwdCertToken },
        ...
}

-- These allow the passing of session keys within the H.245 OLC structure.
-- They are encoded as standalone ASN.1 and based as an OCTET STRING within H.245
H235Key      ::=CHOICE  -- this is used with the H.245 "h235Key" field
{
        secureChannel               KeyMaterial,
        sharedSecret                ENCRYPTED {EncodedKeySyncMaterial},
        certProtectedKey            SIGNED { EncodedKeySignedMaterial },
        ...
}

KeySignedMaterial ::= SEQUENCE {
        generalId           Identifier, -- slave's alias
        mrandom             RandomVal, -- master's random value
        srandom             RandomVal OPTIONAL, -- slave's random value
        timeStamp           TimeStamp OPTIONAL, -- master's timestamp for unsolicited EU
        encrptval           ENCRYPTED {EncodedKeySyncMaterial }
}
EncodedKeySignedMaterial ::= TYPE-IDENTIFIER.&Type (KeySignedMaterial)

H235CertificateSignature ::= SEQUENCE
{
        certificate                 TypedCertificate,
        responseRandom              RandomVal,
        requesterRandom             RandomVal OPTIONAL,
        signature                   SIGNED { EncodedReturnSig },
        ...
}
```

```
ReturnSig ::= SEQUENCE {
        generalId               Identifier, -- slave's alias
        responseRandom          RandomVal,
        requestRandom           RandomVal OPTIONAL,
        certificate             TypedCertificate OPTIONAL -- requested certificate
}

EncodedReturnSig ::= TYPE-IDENTIFIER.&Type (ReturnSig)
KeySyncMaterial    ::= SEQUENCE
{
        generalID           Identifier,
        keyMaterial         KeyMaterial,
        ...
}
EncodedKeySyncMaterial::=TYPE-IDENTIFIER.&Type (KeySyncMaterial)


END   -- End of H235-SECURITY-MESSAGES DEFINITIONS
```

ANNEXE B

Points spécifiques de l'UIT-T H.323

B.1 Rappel

La Figure B.1 donne un aperçu général du domaine d'application de la présente Recommandation dans le cadre de l'UIT-T H.323.

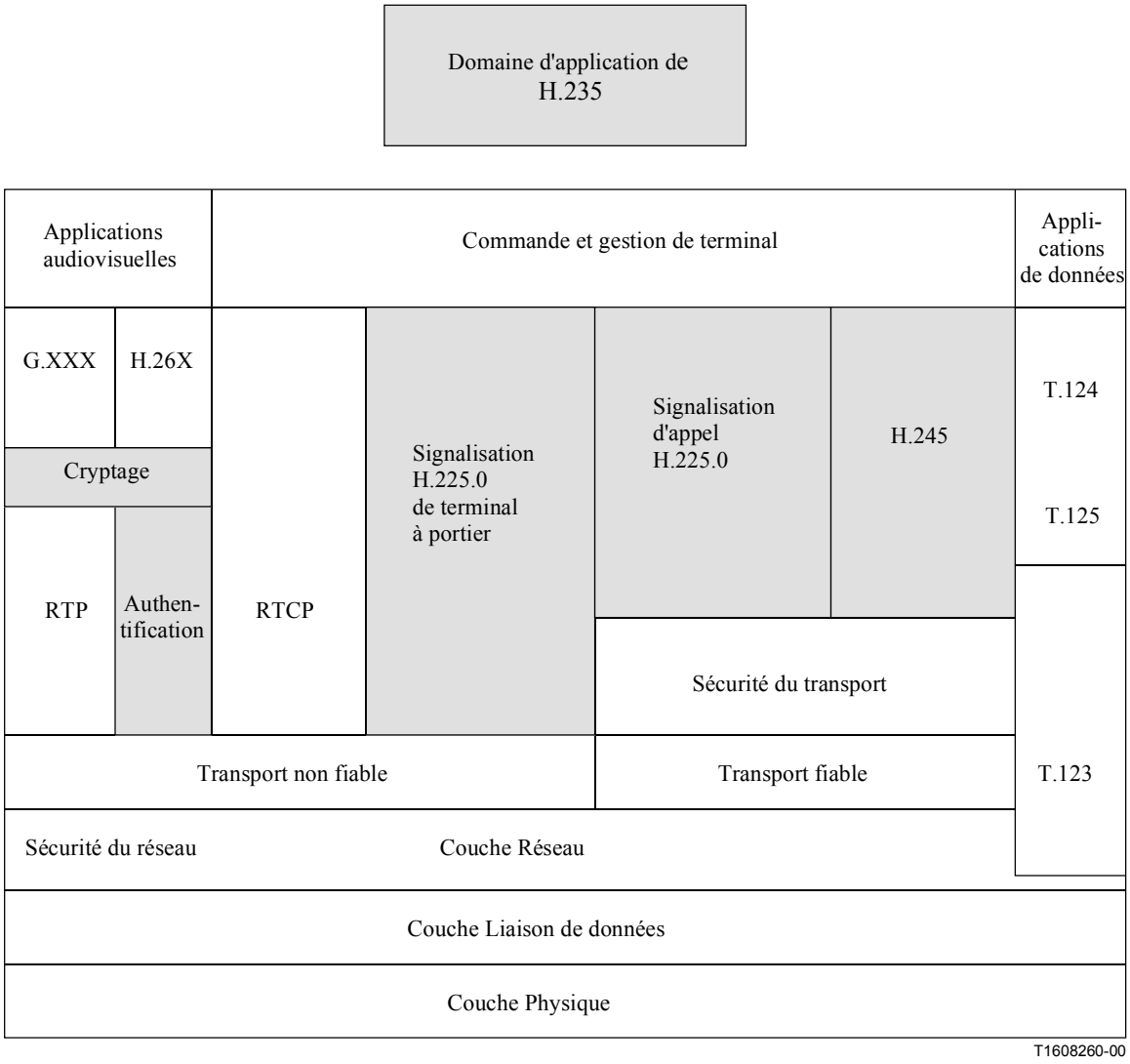


Figure B.1/H.235 – Aperçu général

Pour les flux H.323, la signalisation de l'utilisation du protocole TLS, IPSEC ou d'un mécanisme privé sur le canal de commande H.245 doit être effectuée sur le canal H.225.0 sécurisé ou non sécurisé, pendant l'échange initial de messages Q.931.

B.2 Signalisation et procédures

Les procédures décrites au paragraphe 8/H.323 (procédures de signalisation d'appel) doivent être suivies. Les points d'extrémité H.323 doivent avoir la capacité de coder et de reconnaître la présence (ou l'absence) de prescriptions de sécurité (pour le canal H.245) signalées dans les messages H.225.0.

Si le canal H.225.0 lui-même doit être sécurisé, les mêmes procédures qu'au paragraphe 8/H.323 doivent être suivies. La différence d'exploitation est que les communications ne doivent avoir lieu qu'après connexion à l'identificateur de point TSAP sécurisé et au moyen des modes de sécurité prédéterminés (TLS par exemple). Etant donné que les messages H.225.0 sont les premiers échangés lors de l'établissement de communications H.323, il ne peut pas y avoir de négociation de sécurité "dans la bande" pour les messages H.225.0. En d'autres termes, les deux parties doivent savoir *a priori* qu'elles vont utiliser un mode de sécurité particulier. Pour les flux H.323 en protocole IP, un autre accès notoire (1300) est utilisé pour les communications sécurisées par la méthode TLS.

Un des résultats des échanges H.225.0, dans la mesure où ils concernent la sécurité des flux H.323, est d'offrir un mécanisme permettant d'installer le canal H.245 sécurisé. Facultativement, l'authentification peut se produire pendant l'échange de messages H.225.0. Cette authentification peut être fondée sur des certificats ou sur des mots de passe, avec cryptage et/ou dispersion d'adressage (c'est-à-dire signature). Les particularités de ces modes de fonctionnement sont décrits aux 10.2 à 10.3.4.

Un point d'extrémité H.323 qui reçoit un message "d'établissement" SETUP avec la capacité **h245SecurityCapability** activée doit répondre en indiquant le mode acceptable correspondant (**h245SecurityMode**) dans le message de "connexion" CONNECT. Lorsqu'il n'y a pas de capacités correspondantes, le terminal appelé peut refuser la connexion en envoyant un message **Release Complete** avec le code de cause mis à ***SecurityDenied***. Cette erreur est destinée à n'acheminer aucune information sur une éventuelle discordance de sécurité: le terminal appelant devra déterminer le problème par un autre moyen. Lorsque le terminal appelant reçoit un message de "connexion" sans mode de sécurité suffisant ou acceptable, ce terminal peut mettre fin à l'appel par un message **Release Complete** avec le code de cause ***SecurityDenied***. Lorsque le terminal appelant reçoit un message de "connexion" sans aucune capacité de sécurité, ce terminal peut mettre fin à l'appel par un message **Release Complete** avec la cause ***undefinedReason***.

Si le terminal appelant reçoit un mode **h245Security** acceptable, il doit ouvrir et exploiter le canal H.245 dans le mode de sécurité indiqué. L'échec d'établissement du canal H.245 dans le mode de sécurité déterminé ici doit être considéré comme une erreur de protocole et la connexion doit être fermée.

B.2.1 Compatibilité avec la Révision 1

Un point d'extrémité possédant la capacité de sécurité ne doit pas renvoyer, à un point d'extrémité ne possédant pas la capacité de sécurité, de champs, d'indications ou d'états liés à la sécurité. Si un appelé reçoit un message "d'établissement" qui ne contient pas les capacités de sécurité **H245Security** ni/ou un jeton d'authentification, cet utilisateur peut renvoyer un message **ReleaseComplete** afin de refuser la connexion; mais dans ce cas il doit utiliser le code de cause ***UndefinedReason***. De manière analogue, si un appelant reçoit un message de "connexion" sans indication **H245SecurityMode** et/ou un jeton d'authentification ayant envoyé un message "d'établissement" avec **H245Security** et/ou un jeton d'authentification, cet utilisateur peut également mettre fin à la connexion en émettant un message **ReleaseComplete** avec le code de cause ***UndefinedReason***.

B.3 Liaisons avec les protocoles RTP/RTCP

L'utilisation du cryptage dans un flux en protocole de transport en temps réel RTP suivra la méthode générale qui a été recommandée dans le document indiqué par la commande [RTP]. Le cryptage du média doit être assuré sur une base indépendante, paquet par paquet[1]. L'en-tête RTP (y compris

[1] Il convient de noter que, si la longueur d'un paquet de protocole RTP est supérieure à celle d'une unité MTU, une perte partielle (d'un fragment) provoquera l'indéchiffrabilité de l'ensemble du paquet RTP.

l'en-tête de capacité utile) ne doit pas être crypté. La synchronisation de nouvelles clés et du texte chiffré est fondée sur une capacité utile de type dynamique.

La clé de cryptage initiale est présentée par le maître en même temps que le numéro de capacité utile dynamique (par un message **EncryptionSync** dans un canal UIT-T H.245). Le ou les récepteurs du flux média doit commencer à utiliser la clé dès la réception de ce numéro de capacité utile dans l'en-tête RTP. Une ou plusieurs nouvelles clés peuvent être distribuées à tout moment par le point d'extrémité maître. La synchronisation de la toute nouvelle clé avec le flux média doit être indiquée par la transition du type de capacité utile à une nouvelle valeur dynamique. On notera que les valeurs spécifiques n'ont pas d'importance du moment qu'elle changent à chaque distribution d'une nouvelle clé.

L'on part du principe que le cryptage n'est appliqué qu'à la capacité utile dans chaque paquet RTP; les en-têtes RTP restent en clair. On suppose que tous les paquets RTP sont un multiple entier d'octets. La façon dont les paquets RTP sont encapsulés dans la couche Transport ou Réseau est hors du domaine d'application de la présente Recommandation. Tous les modes doivent prévoir la perte (ou le déclassement) de paquets, ainsi que le bourrage de paquets pour qu'ils comportent un multiple approprié d'octets.

Le décryptage du flux doit être effectué sans tenir compte du contexte des états afin de tenir compte du fait que des paquets peuvent être perdus; chaque paquet doit donc être déchiffrable isolément. Deux exigences du mode algorithmique par blocs doivent s'appliquer comme suit:

a) *vecteurs d'initialisation*

La plupart des modes par blocs impliquent un certain "chaînage"; chaque cycle de cryptage dépend d'une certaine manière de la sortie du cycle précédent. Au début d'un paquet, une certaine valeur initiale de bloc [généralement appelée vecteur d'initialisation (IV, *initialization vector*)] doit donc être fournie afin de commencer le processus de cryptage. Quel que soit le nombre d'octets de flux qui sont traités à chaque cycle de cryptage, la longueur du vecteur d'initialisation est toujours égale à celle d'un bloc. Tous les modes, sauf le mode dictionnaire (ECB, *electronic code book*), nécessitent un vecteur d'initialisation. Dans tous les cas, un vecteur d'initialisation doit être construit à partir des B premiers octets (où B est la longueur de bloc) de la séquence (Seq# + pointeur temporel). Ce motif doit être répété jusqu'à ce qu'un nombre d'octets suffisant ait été produit. Il convient de noter que le vecteur d'initialisation produit de cette façon peut faire apparaître un motif de clé qui est considéré comme "faible" pour un algorithme particulier.

b) *bourrage*

Les modes ECB (dictionnaire électronique) et CBC (chaînage de blocs chiffrants) traitent toujours le flux d'entrée bloc par bloc. Alors que les modes CFB (rebouclage du cryptogramme) et OFB (rebouclage autoclave sur la sortie) peuvent traiter un nombre N ($\leq$ B) quelconque d'octets du flux d'entrée, il est recommandé que N = B.

Deux méthodes permettent de traiter les paquets dont la capacité utile n'est pas un multiple de blocs:

1) l'emprunt cryptographique pour les modes ECB et CBC; le bourrage à zéro pour les modes CFB et OFB;
2) le bourrage de la façon prescrite par la commande [RTP, section 5.1].

Le protocole [RTP, section 5.1] décrit une méthode de bourrage dans laquelle la capacité utile est bourrée jusqu'à un multiple des blocs, le dernier octet indiquant le nombre d'octets de bourrage (y compris ce dernier octet) et le bit P étant activé dans l'en-tête RTP. La valeur du bourrage doit être déterminée par la convention normale de l'algorithme cryptographique.

Toutes les implémentations conformes à la H.235 doivent prendre en charge les deux méthodes. La méthode utilisée peut être déduite comme suit: si le bit P est activé dans l'en-tête RTP, le paquet est bourré; si le paquet n'est pas un multiple de la longueur B et que le bit P ne soit pas activé, la méthode d'extraction cryptographique s'applique. Sinon, le paquet est un multiple de B et le bourrage ne s'applique pas.

La protection du flux RTP contre les atteintes à l'intégrité et les répétitions fera l'objet d'un complément d'étude.

L'application des techniques cryptographiques aux éléments du protocole de commande en temps réel (RTCP) fera l'objet d'un complément d'étude.

B.4 Procédures et signalisation des messages d'enregistrement, admission et état (RAS) pour l'authentification

B.4.1 Introduction

La présente annexe n'indiquera explicitement aucune forme de secret des messages échangés entre portiers et points d'extrémité. Il existe deux types d'authentification pouvant être utilisés. Le premier type est fondé sur un cryptage symétrique ne nécessitant aucun contact préalable entre le point d'extrémité et le portier. Le deuxième type est fondé sur un abonnement et aura deux formes: mot de passe ou certificat. Toutes ces formes sont issues des procédures indiquées aux 10.1, 10.2.2, 10.2.3 et 10.2.4. Dans la présente annexe, les étiquettes génériques (des points EPA et EPB), indiquées dans les paragraphes précédents, représenteront respectivement le point d'extrémité et le portier.

B.4.2 Authentification entre point d'extrémité et portier (non fondée sur abonnement)

Ce mécanisme peut fournir au portier une indication cryptographique selon laquelle un point d'extrémité particulier, qui s'est préalablement enregistré, est bien celui qui émettra les messages RAS ultérieurs. Il y a lieu de noter que ce procédé peut ne pas fournir au point d'extrémité une quelconque authentification du portier, à moins que l'élément facultatif de signature soit inclus. L'établissement de la relation d'identité s'effectue lorsque le terminal émet la demande **GRQ**, comme indiqué au 7.2.1/H.323. L'échange selon la méthode Diffie-Hellman s'effectue conjointement avec les messages **GRQ** et **GCF**, comme indiqué dans la première phase du 10.1. Cette clé à secret partagé doit ensuite être utilisée pour toute demande **RRQ/URQ** subséquente, envoyée par le terminal au portier. Si un portier fonctionne dans ce mode et reçoit une demande **GRQ** sans jeton contenant la valeur *DHset* ou une valeur algorithmique acceptable, ce portier doit renvoyer, dans le message de rejet **DRJ**, le code de cause **securityDenial**.

La clé à secret partagé qui a été créée par la méthode Diffie-Hellman au cours de l'échange de messages **GRQ/GCF** peut être utilisée pour l'authentification dans d'ultérieures messages de type **xRQ**. Les procédures suivantes doivent être utilisées pour réaliser ce mode d'authentification.

Terminal (**xRQ**):

1) le terminal doit fournir toutes les informations contenues dans le message, comme décrit dans les paragraphes appropriés de l'UIT-T H.225.0;
2) le terminal doit chiffrer l'identificateur du portier **GatekeeperIdentifier** (renvoyé dans le message **GCF**) au moyen de la clé à secret partagé qui a été négociée. Ce cryptogramme doit être transmis dans un jeton cryptographique **clearToken** (voir 10.2) en tant qu'identificateur général **generalID**.

Les 16 bits du nombre aléatoire **random** puis du numéro **requestSeqNum** doivent être combinés par un opérateur OUX avec chacun des 16 bits de l'identificateur de portier **GatekeeperIdentifier**. Si cet identificateur **GatekeeperIdentifier** ne se termine pas par une limite paire à la 16e position, les 8 derniers éléments binaires de l'identificateur de portier **GatekeeperIdentifier** doivent être combinés par un opérateur OUX avec l'octet de plus faible poids de la valeur aléatoire puis avec le

numéro **requestSeqNum**. L'identificateur de portier **GatekeeperIdentifier** doit être chiffré au moyen de l'algorithme sélectionné dans le message **GCF** (algorithmOID) et au moyen de l'ensemble du secret partagé.

Les exemples suivants donnent un aperçu de cette procédure:

RND16: valeur à 16 bits de la valeur aléatoire

SQN16: valeur à 16 bits du numéro requestSeqNum

BMPX: le X^e caractère BMP de l'identificateur GatekeeperIdentifier

BMP1' = (BMP1) XOR (RND16) XOR (SQN16)

BMP2' = (BMP2) XOR (RND16) XOR (SQN16)

BMP3' = (BMP3) XOR (RND16) XOR (SQN16)

BMP4' = (BMP4) XOR (RND16) XOR (SQN16)

BMP5' = (BMP5) XOR (RND16) XOR (SQN16)

:

:

BMPn' = (BMPn) XOR (RND16) XOR (SQN16)

Afin de relier cryptographiquement ce message et les messages ultérieurs avec l'entité qui s'est enregistrée initialement (le point d'extrémité qui a émis la demande **RRQ**), la plus récente valeur aléatoire **random** renvoyée doit être utilisée (cette valeur peut être plus récente que celle qui a été renvoyée dans le message **RCF** faisant suite à un message **xCF** ultérieur).

Portier (**xCF/xRJ**)

1) le portier doit chiffrer son identificateur **GatekeeperIdentifier**(conformément à la procédure ci-dessus) avec la clé à secret partagé qui est associée à l'alias du point d'extrémité; il doit ensuite le comparer à la valeur contenue dans la demande **xRQ**;
2) le portier doit renvoyer un message de rejet **xRJ** si les deux valeurs chiffrées ne correspondent pas;
3) si son identificateur **GatekeeperIdentifier** correspond à la valeur demandée, le portier doit appliquer toute logique locale éventuelle puis répondre par un message **xCF** ou **xRJ**;
4) si un message **xCF** est envoyé par le portier, ce message doit contenir un identificateur de point d'extrémité **EndpointIdentifier** assigné et une nouvelle valeur aléatoire dans le champ **random** d'un paramètre **clearToken**.

Pour la représentation graphique de cet échange, voir la phase 2 de la Figure 1. Le portier connaît la clé à secret partagé qu'il faut utiliser pour déchiffrer l'identificateur de portier indiqué dans le nom d'alias du message.

B.4.3 Authentification entre point d'extrémité et portier (fondée sur abonnement)

Tous les messages RAS autres que GRQ/GCF doivent normalement contenir les jetons d'authentification requis par le mode de fonctionnement spécifique. Il existe trois variantes différentes qui peuvent être mise en œuvre, selon les exigences et l'environnement:

1) authentification par mot de passe avec cryptage symétrique;
2) authentification par mot de passe avec hachage;
3) authentification par certificat avec signatures.

Dans tous les cas, le jeton contiendra les informations décrites dans les sous-paragraphes suivants, selon la variante choisie. Si un portier fonctionne en mode sécurisé et reçoit un message RAS sans valeur de jeton acceptable, il doit renvoyer un code de cause **securityDenial** dans le message de

rejet. Dans tous les cas, le jeton renvoyé par le portier est facultatif: s'il est omis, seule une authentification à sens unique est effectuée.

B.4.3.1 Mot de passe avec cryptage symétrique

La phase de découverte du portier (GRQ, GCF et GRJ) peut échouer comme indiqué sur la Figure B.2, ou au contraire aboutir, au moyen du paramètre **cryptoTokens**.

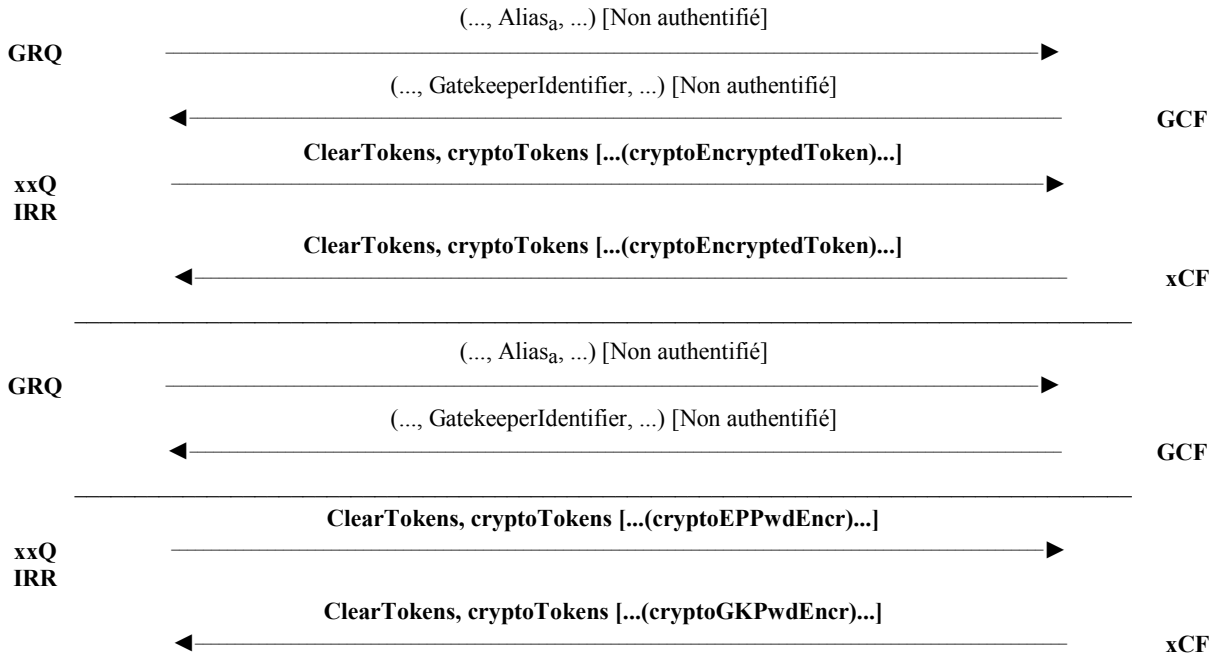


Figure B.2/H.235 – Mot de passe avec cryptage symétrique

B.4.3.2 Mot de passe avec hachage

La phase de découverte du portier (GRQ, GCF et GRJ) peut échouer comme indiqué sur la Figure B.3, ou au contraire aboutir, conformément à l'Annexe D, au moyen du paramètre **cryptoTokens**.

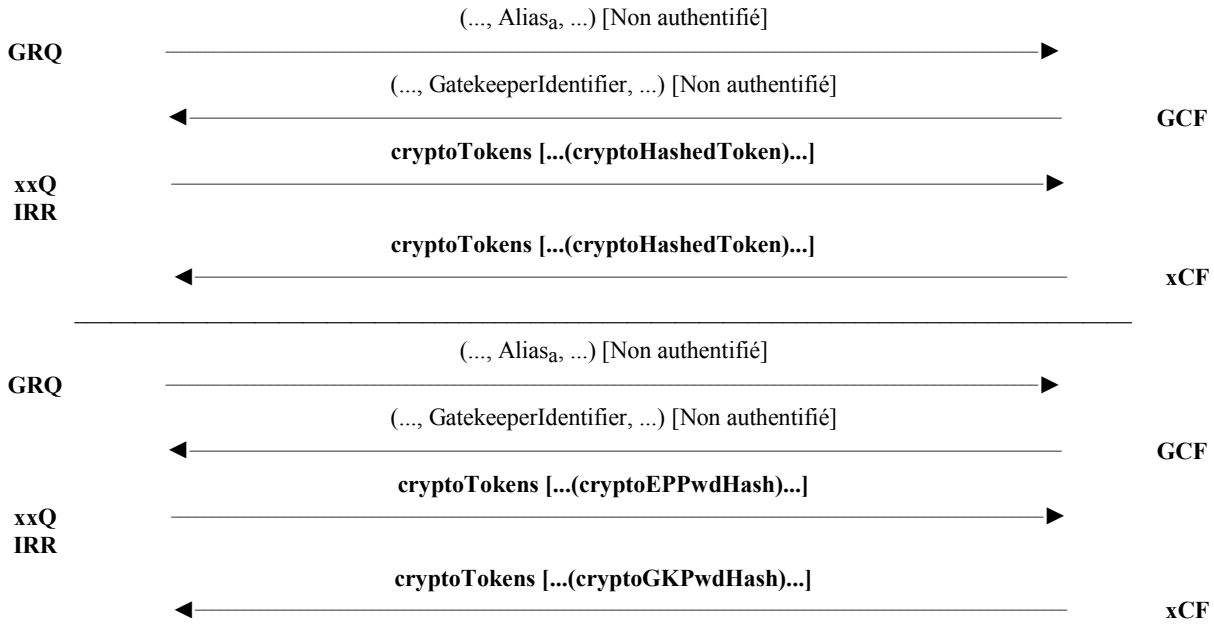


Figure B.3/H.235 – Mot de passe avec hachage

B.4.3.3 Authentification par certificat avec signatures

La phase de découverte du portier (GRQ, GCF et GRJ) peut échouer comme indiqué sur la Figure B.4, ou au contraire aboutir, conformément à l'Annexe E, au moyen du paramètre **cryptoTokens**.

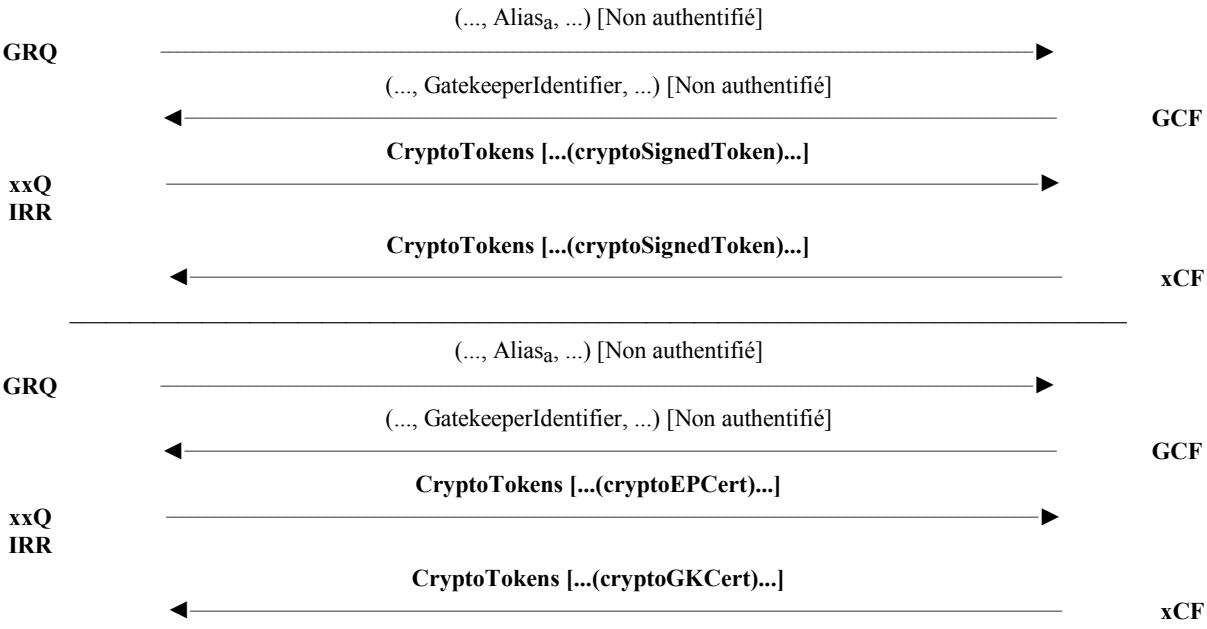


Figure B.4/H.235 – Authentification par certificat avec signatures

B.5 Interactions non terminales

B.5.1 Passerelle

Comme indiqué au 6.6, une passerelle H.323 doit être considérée comme un élément crédibilisé. Cela comprend les passerelles entre protocoles (H.323-H.320, etc., ...) et les passerelles de sécurité (serveurs tampons/pare-feu). Le secret des communications multimédias peut être assuré entre le point d'extrémité communicant et la passerelle tête de ligne. Mais ce qui se produit au-delà de la passerelle doit être considéré *a priori* comme non sécurisé.

ANNEXE C

Points spécifiques de l'UIT-T H.324

A étudier.

ANNEXE D

Profil de sécurité élémentaire

D.1 Introduction

La présente annexe définit des profils de sécurité élémentaires simples, basés sur le profil de sécurité UIT-T H.235 et ceux que proposent actuellement l'ETSI et l'IMTC. Ces profils font appel à des caractéristiques de sécurité appropriées tirées de l'UIT-T H.235 et de son vaste choix d'options.

D.2 Conventions de spécification

Quelques explications sont nécessaires pour la compréhension des termes utilisés:

la présente annexe définit un **profil de sécurité élémentaire**, qui fournit la sécurité élémentaire par des moyens simples utilisant des techniques cryptographiques de type à mot de passe. Au besoin, ce profil élémentaire peut utiliser le **profil de sécurité à cryptage vocal** pour assurer la confidentialité de la parole. On trouvera dans l'Annexe E un profil de sécurité plus élaboré à signatures numériques qui n'est pas exposé aux limitations du profil élémentaire.

La présente annexe utilise des champs H.235 pour fournir des services de sécurité de type authentification/intégrité au moyen de messages de signalisation H.323. Divers identificateurs d'objet (voir D.11) déterminent le service de sécurité qui est effectivement sélectionné et la version de protocole de la présente Recommandation qui est en cours d'utilisation. La procédure I spécifie la manière de mettre en œuvre les services de sécurité au moyen de certains mécanismes de sécurité tels que les techniques symétriques (hachage sur clés calculées). Les identificateurs d'objet sont désignés au moyen d'une référence symbolique dans le texte (par exemple "A").

Si le service d'intégrité du message fournit toujours l'authentification du message, l'inverse n'est pas nécessairement vrai. Dans la pratique, l'authentification et le service d'intégrité combinés exploitent les mêmes données relatives aux clés sans introduire de faiblesse au niveau de la sécurité.

De plus, toutes les informations de sécurité relatives aux bonds individuels sont mises dans l'élément **CryptoHashedToken**. Ces informations sont recalculées à chaque bond.

Généralement, le mot de passe, la clé de session et le secret partagé ont en commun qu'ils sont utilisés en cryptographie symétrique entre deux (ou plusieurs) entités. La différence entre un mot de passe et une clé de session/secret partagé est la manière dont les clés sont effectivement appliquées, par exemple les mots de passe pour l'authentification et l'autorisation, les clés de session pour le cryptage. Le terme secret partagé est relativement neutre étant donné qu'il ne se réfère pas à un usage spécifique.

Le **mot de passe** (que l'on peut assimiler à un secret partagé) est utilisé pour l'authentification/intégrité des messages RAS et H.225.0 étant donné que cet élément peut être introduit par l'utilisateur. Généralement, le mot de passe a une durée de vie plus longue. Il est connu a priori et peut-être défini dans le contexte du processus global d'abonnement de l'utilisateur. Certains algorithmes (par exemple, le passage des mots de passe dans un algorithme de hachage) peuvent transformer le mot de passe afin de faciliter son traitement dans les protocoles et d'aboutir à une longueur fixe.

La **clé de session** utilisée pour le cryptage des flux médias est, quant à elle, produite par l'entité maîtresse pour une session avec un protocole RTP spécifique (sur une structure OLC), pour la durée de la communication au maximum. La clé de session produite est cryptée au moyen d'une clé obtenue à partir du **secret partagé** Diffie-Hellman agréé que les deux points d'extrémité ont calculée. Dans ce cas, le secret partagé DH agit comme une clé maîtresse pour la protection de la ou des clés de session.

Le jeton **ClearToken** H.235 offre un champ destinataire **random** qui contient un entier de 32 bits. Ce champ est utilisé dans le sens suivant: **random** est un nombre croissant monotone qui commence à n'importe quelle valeur et qui augmente à chaque message sortant. Le champ **random** est utilisé comme une valeur de "randomisation" additionnelle pour l'entrée dans la fonction de hachage à dispersion sur clé calculée au cas où plusieurs messages sont émis rapidement les uns après les autres avec des horodateurs identiques. Cela peut se produire lorsque la résolution de l'horloge UTC est insuffisante. En substance, la valeur de hachage produite ou la valeur de contrôle de l'intégrité se distingue par les changements de la valeur de **random**. Ceci a pour but de contrer les agressions par répétition. Pour les besoins de simplicité de la mise en œuvre on préfère, dans le cas présent, un compteur progressif à une séquence réellement aléatoire. Le destinataire peut conserver des couples

timestamp/random reçus au cours de la période définie par une fenêtre[2] temporelle locale. Le même couple **timestamp/random** survenant deux fois signale une agression par répétition.

Le présent profil consiste à "mettre l'identificateur **generalID** de **ClearToken** à l'identificateur du destinataire". Cela signifie en fait que pour des messages RAS, il s'agit du portier GK ou de l'identificateur[3] du point d'extrémité, alors que pour les messages de signalisation d'appel H.225.0, il s'agit de l'identificateur du point d'extrémité destinataire. L'identificateur **sendersID** doit être mis à la chaîne d'identification de l'expéditeur.

Un **block** se réfère à l'unité de base de bits en paquet que le chiffrement par bloc est capable de crypter/décrypter au moyen d'une opération cryptographique élémentaire; dans le cas des normes DES et DES triple, la taille du bloc est de 64 bits.

Pour ne pas devoir citer des marques (telles que RC2®), la présente annexe parlera d'un algorithme de cryptage "compatible RC2".

La présente Recommandation contient des termes bien connus relatifs à la sécurité tels que clé, gestion de clés et dispositif SET, qui ont des sens différents dans d'autres contextes (par exemple clavier tactile, gestion des touches de fonction Q.931/Q.932 et protocoles de transaction électronique sécurisée).

D.3 Domaine d'application

La présente annexe traite de la sécurité des dispositifs d'extrémité simples H.323. Le profil de sécurité peut être appliqué par des terminaux H.323 sécurisés, y compris le **terminal téléphonique simple sécurisé** (SAT, *secure audio simple endpoint type*) défini dans la présente annexe (voir D.6); le profil de sécurité peut être appliqué par d'autres entités H.323 telles que les passerelles, les portiers et les ponts MCU.

D.4 Abréviations

BES service spécialisé (*back-end service*)

CBC mode chaînage de blocs chiffrants (*cipher block chaining*)

DES norme de cryptage des données (*data encryption standard*)

DH Diffie-Hellman

ECB mode dictionnaire (*electronic code book*)

EP point d'extrémité (*endpoint*)

ETSI European Telecommunications Standards Institute

GK portier (*gatekeeper*)

HMAC code d'identification de message avec hachage (*hashed message authentication code*)

IMTC International Multimedia Teleconferencing Consortium

IPSEC sécurité de protocole Internet (*Internet protocol security*)

IV vecteur d'initialisation (*initialization vector*)

MAC code d'identification de message (*message authentication code*)

MD5 compilation de messages 5 (*message digest 5*)

---

[2] La fenêtre temporelle compense les écarts de l'heure symchronisée et les temps de transit dans le réseau.

[3] Qui dépend du sens EP à GK ou inversement.

OID identificateur d'objet (*object identifier*)

PFS confidentialité totale vers l'avant (*perfect forward secrecy*)

RAS enregistrement, admission et statut (*registration, admission and status*)

RSA Rivest, Shamir et Adleman

RTP protocole en temps réel (*real-time protocol*)

SASET dispositif d'extrémité simple audio sécurisé (*secure audio simple endpoint type*)

SET dispositif d'extrémité simple (*simple endpoint type*)

SHA algorithme de hachage sécurisé (*secure hash algorithm*)

TCP protocole de commande de transmission (*transmission control protocol*)

TIPHON harmonisation de protocole de télécommunications et Internet entre réseaux (*telecommunications and Internet protocol harmonization over networks*)

TLS sécurité de la couche Transport (*transport layer security*)

UIT Union internationale des télécommunications

VoIP téléphonie IP (*voice over Internet protocol*)

D.5 Références normatives

DES [FIPS-46-2] US National Bureau of Standards, "Data Encryption Standard", Federal Information Processing Standard, (FIPS) Publication 46-2, décembre 1993, http://www.itl.nist.gov/div897/pubs/fip46-2.htm.

[FIPS-74] US National Bureau of Standards, "Guidelines for Implementing and Using the Data Encryption Standard", Federal Information Processing Standard (FIPS) Publication 74, avril 1981, http://www.itl.nist.gov/div897/pubs/fip74.htm.

[FIPS-81] US National Bureau of Standards, "DES Modes of Operation", Federal Information Processing Standard (FIPS) Publication 81, décembre 1980, http://www.itl.nist.gov/div897/pubs/fip81.htm.

[ISO/CEI 10118-3] *Technologies de l'information – Techniques de sécurité – Fonctions de brouillage – Partie 3: Fonctions de hachage dédiées*, 1998.

[H.225.0] UIT-T H.225.0 Version 2, *Protocoles de signalisation d'appel et mise en paquets des trains multimédias dans les systèmes de communications multimédia en mode paquet*, 1998.

[H.235v1] UIT-T H.235 Version 1, *Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)*, 1998.

[H.235v2] UIT-T H.235 Version 2, *Sécurité et cryptage des terminaux multimédias de la série H (terminaux H.323 et autres terminaux de type H.245)*, 2000.

[H.245] UIT-T H.245 Version 7, *Protocole de commande pour communications multimédias*, 2000.

[H.323] UIT-T H.323 Version 4, *Systèmes de communications multimédias en mode paquet*, 2000.

[Annexe F/H.323] UIT-T H.323 Annexe F, *Dispositifs d'extrémité simples*, 1999.

[RFC 2268] RIVEST (R.): *A Description of the RC2® Encryption Algorithm*, RFC 2268, mars 1998.

D.6 Profil de sécurité élémentaire

Le présent paragraphe propose une base pour le profil de sécurité simple.

D.6.1 Aperçu général

Le profil de sécurité élémentaire utilise le modèle à acheminement par portier. La sécurité élémentaire est applicable dans les environnements gérés ayant des clés symétriques/mots de passe attribués aux entités (terminal à portier, portier à portier, passerelle à portier).

Les caractéristiques proposées par ces profils sont:

– Pour les messages RAS, H.225.0 et H.245:
  - l'authentification de l'utilisateur jusqu'à l'entité voulue, indépendamment du nombre de bonds[4] du niveau application effectués par le message;
  - l'intégrité du message de signalisation proprement dit, y compris les parties (champs) déterminantes des messages parvenant à une entité, indépendamment du nombre de bonds au niveau application qu'effectue le message;
  - l'authentification des messages de signalisation bond par bond au niveau application; l'intégrité offre ces services de sécurité pour l'ensemble du message.

– Pour le flux média:
  - la confidentialité du flux média est obtenue par cryptage asymétrique.

Plusieurs types d'agression sont combattus au moyen des services de sécurité ci-dessus, utilisés de manière appropriée. Il s'agit:

- des agressions visant la fonction de refus de service: une vérification rapide des valeurs de hachage cryptographique peuvent préserver de telles agressions;
- des agressions par entremetteur: l'authentification et l'intégrité des messages bond par bond au niveau application empêchent de telles agressions lorsque l'entremetteur, un routeur hostile par exemple, se trouve entre deux bonds au niveau application;
- des agressions par répétition: l'emploi des horodateurs et des numéros de séquence empêche de telles agressions;
- des mystifications: l'authentification de l'utilisateur empêche de telles agressions;
- du piratage des connexions: l'authentification/intégrité de chaque message de signalisation empêche de telles agressions;
- des écoutes clandestines du flux média, qui sont enrayées par le cryptage et l'utilisation de clés secrètes.

D'autres aspects importants du profil de sécurité simple sont:

- l'emploi d'algorithmes robustes, réputés et largement utilisés, fondés sur les travaux de l'IMTC/ETSI/IETF;
- la capacité de mise en place par niveaux en fonction des besoins de sécurité du modèle commercial;
- l'applicabilité à divers scénarios de mise en place, par exemple les groupes fermés, les environnements échelonnables et les conférences multipoints.

---

[4] Dans le présent contexe, bond a le sens d'un élément de réseau H.235 de confiance (tel que portier, passerelle, pont MCU, serveur mandataire, pare-feu). En conséquence, la sécurité bond par bond au niveau application, lorsqu'elle est utilisée avec des techniques symétriques, n'assure pas une sécurité de bout en bout réelle entre les terminaux.

Le Tableau D.1 groupe par profil de sécurité toutes les procédures définies dans la présente annexe afin de traiter des divers besoins en matière de sécurité. Il englobe le profil de sécurité élémentaire (hachage vertical – en bleu dans la copie électronique) et le profil de sécurité de cryptage vocal (hachage horizontal – en vert dans la copie électronique).

Tableau D.1/H.235 – Résumé des profils de sécurité de l'Annexe D

<table>
<tr><th rowspan="2">Services de sécurité</th><th colspan="6">Fonctions d'appel</th></tr>
<tr><th>RAS</th><th colspan="2">H.225.0</th><th>H.245[a)]</th><th colspan="2">RTP</th></tr>
<tr><td>Authentification</td><td>Mot de passe HMAC-SHA1-96</td><td colspan="2">Mot de passe HMAC-SHA1-96</td><td>Mot de passe HMAC-SHA1-96</td><td colspan="2"></td></tr>
<tr><td>Non-répudiation</td><td></td><td colspan="2"></td><td></td><td colspan="2"></td></tr>
<tr><td>Intégrité</td><td>Mot de passe HMAC-SHA1-96</td><td colspan="2">Mot de passe HMAC-SHA1-96</td><td>Mot de passe HMAC-SHA1-96</td><td colspan="2"></td></tr>
<tr><td>Confidentialité</td><td></td><td colspan="2"></td><td></td><td>DES à 56 bits
56 bits compa-tibles RC2
DES triple à 168 bits</td><td></td></tr>
<tr><td>Contrôle d'accès</td><td></td><td colspan="2"></td><td></td><td colspan="2"></td></tr>
<tr><td>Gestion de clés</td><td>Attribution de mot de passe à la prise d'abonnement</td><td>Attribution de mot de passe à la prise d'abonnement</td><td>Echange de clés Diffie-Hellman authenti-fiées</td><td>Gestion de clés de session H.235 intégrée (distribution de clé, mise à jour de clé par DES à 56 bits compatibles RC2/DES triple à 168 bits)</td><td colspan="2"></td></tr>
<tr><td colspan="7">[a)] En tunnel H.245 ou H.245 incorporé dans une connexion rapide H.225.0.</td></tr>
</table>

Pour les besoins d'authentification, l'utilisateur doit utiliser un système à mot de passe, système fortement recommandé pour l'authentification en raison de sa simplicité et de sa facilité de mise en œuvre. Le hachage de tous les champs des messages H.225.0 est la méthode recommandée pour aboutir à l'intégrité des messages (utilisant également le système à mot de passe).

Les entités H.323 sécurisées au moyen de ce profil de sécurité effectuent l'authentification conjointement avec l'intégrité au moyen du même mécanisme de sécurité commun.

Pour ce qui concerne la confidentialité vocale, facultative, le système suggéré est le cryptage par norme compatible RC2, DES ou DES triple fondé sur le modèle commercial et les besoins d'exportabilité. Certains environnements qui offrent déjà un degré donné de confidentialité ne nécessitent pas expressément le cryptage vocal. Si c'est le cas, la concordance des clés Diffie-Hellman et les autres procédures de gestion des clés sont également superflues.

Lorsqu'elles mettent en œuvre le profil de sécurité de cryptage vocal, les entités H.323 doivent appliquer la norme DES à 56 bits en tant qu'algorithmes de cryptage par défaut; elles peuvent aussi utiliser la norme DES triple à 168 bits ainsi que le cryptage exportable au moyen d'une norme compatible RC2 à 56 bits.

Les moyens de contrôle d'accès ne sont pas décrits explicitement; ils peuvent être mis en œuvre localement compte tenu de l'information reçue acheminée dans les champs de signalisation H.235 (jeton ClearToken, jeton CryptoToken).

La présente Recommandation ne décrit pas les procédures avec la direction et l'administration pour l'attribution des mots de passe/clés secrètes à la souscription à l'abonnement. De telles procédures peuvent être exécutées par des moyens qui ne sont pas traités dans la présente annexe.

Les entités de communication concernées ont la possibilité de déterminer implicitement l'usage qu'elles feront du profil de sécurité élémentaire ou du profil de sécurité à signature par l'évaluation des identificateurs d'objet de sécurité signalés dans les messages (identificateurs **tokenOID** identificateurs, **algorithmOID**; voir également D.11).

D.6.1.1 Profil de sécurité élémentaire

Le profil de sécurité élémentaire est applicable dans un environnement où des mots de passe/clés symétriques auxquels a souscrit l'abonné peuvent être attribués aux entités H.323 sécurisées (terminaux, etc.) et aux éléments de réseau (portiers, serveurs mandataires). Il offre l'authentification et l'intégrité pour les messages RAS, H.225.0 et H.245 canalisés en tunnel au moyen du hachage HMAC-SHA1-96 à mot de passe et tel que spécifié par la procédure I. L'établissement de la communication H.225.0 au moyen de FastStart (portier à portier ou terminal à terminal) englobe la gestion de clés Diffie-Hellman intégrée.

La zone hachurée verticalement (en bleu dans la copie électronique) du Tableau D.2 représente le profil de sécurité élémentaire.

Tableau D.2/H.235 – Profil de sécurité élémentaire

Services de sécurité	Fonctions d'appel			
	RAS	**H.225.0**	**H.245**	**RTP**
Authentification et intégrité	Mot de passe HMAC-SHA1-96	Mot de passe HMAC-SHA1-96	Mot de passe HMAC-SHA1-96	
Non-répudiation				
Confidentialité				
Contrôle d'accès				
Gestion de clés	Attribution de mots de passe à la souscription à l'abonnement	Attribution de mots de passe à la souscription à l'abonnement		

Facultativement, le profil de sécurité de cryptage vocal peut être aisément combiné avec le profil de sécurité élémentaire. Il est possible de crypter les flux audio au moyen du profil de sécurité de cryptage vocal faisant appel à une norme DES, compatible RC2 ou DES triple ainsi que l'utilisation de la procédure d'échange de clés Diffie-Hellman authentifié.

Le profil de sécurité élémentaire autorise la procédure de connexion rapide avec des éléments de gestion de clés intégrés. Des moyens de signalisation sont également fournis pour l'actualisation de la clé et la synchronisation des messages H.245 canalisés en tunnel. Dans le cas des appels de longue durée, ces messages H.245 nécessitent la canalisation en tunnel dans des messages H.225.0.

D.6.1.2 Profil de sécurité de cryptage vocal

Le profil de sécurité de cryptage vocal n'est pas un profil indépendant comme c'est le cas du profil de sécurité élémentaire; il s'agit plutôt d'une option du profil de sécurité susmentionné qui peut être utilisée en association avec celui-ci. Ce profil table également sur certains services de sécurité dans le cadre de la signalisation d'appel et des procédures d'établissement de la connexion, par exemple la concordance de clés Diffie-Hellman et d'autres fonctions de la gestion des clés.

Les entités H.323 peuvent mettre en œuvre le profil de cryptage vocal pour obtenir la confidentialité vocale. Trois algorithmes de cryptage sont proposés: le système proposé est le système crypté au moyen d'une norme compatible RC2, de la norme DES ou DES triple fondée sur le modèle commercial et le besoin d'exportabilité. Quelques environnements qui offrent déjà un certain degré de confidentialité ne nécessitent éventuellement pas le cryptage vocal. Si c'est le cas, la concordance de clés Diffie-Hellman et d'autres procédures de gestion des clés sont également superflues.

Lorsque des entités H.323 utilisent le cryptage vocal, le profil de sécurité doit implémenter la norme DES à 56 bits en tant qu'algorithme de cryptage par défaut; elles peuvent aussi utiliser la norme DES triple à 168 bits ainsi que le cryptage exportable utilisant la norme compatible RC2 à 56 bits.

Le profil de cryptage vocal est défini au paragraphe D.2.

Tableau D.3/H.235 – Profil de cryptage vocal

<table>
<tr><th>Services de sécurité</th><th colspan="5">Fonctions d'appel</th></tr>
<tr><th></th><th>RAS</th><th>H.225.0</th><th>H.245</th><th colspan="3">RTP</th></tr>
<tr><td>Authentification et intégrité</td><td></td><td></td><td></td><td colspan="3"></td></tr>
<tr><td>Non-répudiation</td><td></td><td></td><td></td><td colspan="3"></td></tr>
<tr><td>Confidentialité</td><td></td><td></td><td></td><td>DES à 56 bits</td><td>Compatible RC2 à 56 bits</td><td>DES triple à 168 bits</td></tr>
<tr><td>Contrôle d'accès</td><td></td><td></td><td></td><td colspan="3"></td></tr>
<tr><td>Gestion de clés</td><td></td><td>Echange de clés Diffie-Hellman authentifié</td><td>Gestion de clés H.235 intégrée (distribution de clé, mise à jour de clé)</td><td colspan="3"></td></tr>
</table>

D.6.2 Authentification et intégrité

La présente annexe utilise les termes suivants dans le contexte des services de sécurité:

- **authentification et intégrité**: partie de service de sécurité combinée du profil élémentaire qui prend en charge l'intégrité du message en association avec l'authentification de l'utilisateur. Celui-ci pourrait obtenir l'authentification en appliquant correctement une clé secrète partagée. Les deux services de sécurité sont assurés par le même mécanisme de sécurité.

Lorsque l'on utilise deux techniques de clés symétriques, l'authentification/intégrité des services de sécurité s'applique uniquement au niveau bond par bond.

D.6.3 Prescriptions H.323

Les entités H.323 qui implémentent ce profil de sécurité élémentaire sont supposées prendre en charge les prescriptions H.323 suivantes:

- la connexion rapide;
- le modèle à routage par portier.

D.6.3.1 Aperçu général

Description de la procédure à utiliser dans ce profil.

La procédure I est un mécanisme d'authentification de messages de signalisation de type à clés symétriques simple basé sur un mot de passe connu par deux entités (par exemple un portier et un point d'extrémité H.323). Cette procédure assure l'authentification et l'intégrité des messages RAS, Q.931 et H.245 (voir D.6.3.2).

Selon la politique de sécurité, l'authentification peut être unilatérale ou réciproque et s'appliquer à l'authentification/intégrité en sens inverses comme elle peut assurer par la même occasion une sécurité plus élevée. Le portier décide s'il y a lieu d'appliquer également l'authentification/l'intégrité dans le sens opposé.

Les portiers qui détectent un échec de validation de l'authentification et/ou de l'intégrité dans un message RAS ou un message de signalisation d'appel provenant d'un point d'extrémité sécurisé ou d'un portier homologue répondent au moyen d'un message de rejet correspondant indiquant l'absence de sécurité par la mise du motif de refus à **securityDenial**.

Il existe une signalisation H.235 implicite pour indiquer l'utilisation de la procédure I et du mécanisme de sécurité appliqué basée sur la valeur des identificateurs d'objet (voir également D.11) et sur le contenu des champs de message.

Le présent profil n'utilise pas les champs ICV H.235; en effet, les valeurs de contrôle d'intégrité cryptographique sont traitées comme des valeurs de hachage cryptographique et sont mises dans les champs de hachage de **CryptoToken**.

D.6.3.2 Détails de l'authentification des messages de signalisation de type à clés symétriques (Procédure I)

Il faudra suivre les procédures ci-après en cas d'emploi de la procédure I:

- une valeur de hachage de 12 octets (96 bits) est utilisée avec les algorithmes HMAC SHA1-96 pour produire un authentificateur. Pour produire la clé à partir d'un mot de passe, il *faut* utiliser le mécanisme décrit au 10.3.5.

  NOTE 1 – Lorsqu'on détermine la clé secrète à partir d'un mot de passe entré par l'utilisateur, il faut lui conférer un caractère aléatoire suffisant. Il est recommandé, par exemple, d'utiliser des secrets réellement aléatoires pour la clé secrète ou de s'assurer que les mots de passe aléatoires sont suffisamment longs.

- Le champ **CryptoH323Token** de chaque message RAS/H.225.0 doit contenir les champs suivants:
  - **nestedCryptoToken** contenant un **CryptoToken** contenant à son tour **cryptoHashedToken** avec les champs suivants:
    - **tokenOID** mis à "A" pour indiquer que le calcul d'authentification/intégrité porte sur tous les champ du message RAS/H.225.0.
    - **hashedVals** contenant le champ **ClearToken** utilisé avec les champs suivants:
      - **tokenOID** mis à "T" indiquant que **ClearToken** est en cours d'utilisation pour l'authentification/intégrité de message
      - **timeStamp** contenant l'horodateur
      - **random** contenant un numéro d'ordre croissant monotone. Ce nombre permet de conférer l'unicité individuelle à deux messages ayant le même horodateur (dans les limites de la résolution d'horloge)
      - **generalID** contenant l'identificateur du destinataire (uniquement dans le cas de messages monodiffusés)
      - **sendersID** contenant l'identificateur de l'expéditeur
      - **dhkey**, utilisé pour transférer les paramètres Diffie-Hellman comme spécifié dans la présente Recommandation pendant **Setup** et **Connect**

- **halfkey** contenant la clé publique aléatoire d'une des parties;
- **modsize** contenant DH-prime (voir Tableau D.4);
- **generator** contenant DH-group (voir Tableau D.4).

NOTE 2 – Lorsque le profil de sécurité élémentaire est utilisé sans le profil de sécurité à cryptage vocal, aucun paramètre Diffie-Hellman ne doit être envoyé; au lieu de cela, **halfkey**, **modsize** et **generator** peuvent être mis à la représentation binaire de 0 pour des raisons de simplicité.

– **token** contenant **HASHED,** avec les champs:
  - **algorithmOID** mis à "U" pour indiquer l'utilisation du code HMAC-SHA1-96
  - **params** mis à NULL
  - **hash** contenant l'authentificateur calculé au moyen du code HMAC-SHA1-96. L'authentificateur peut être calculé
    – sur l'ensemble des champs RAS/H.225.0 du message si **tokenOID** de **CryptoHashedToken** est mis à "A" (indiquant l'authentification et l'intégrité).

tokenOID est mis à "A" pour la protection des unités H323-UU-PDU canalisées en tunnel, y compris tous les contenus des messages H.245; le calcul du hachage doit porter sur l'ensemble des messages **H.225.0 PDU**, tous les champs étant conformes à la procédure décrite au D.6.3.3.2.

- L'authentificateur est vérifié à la fin de chaque dernier tronçon de canal (point d'extrémité 1 à portier 1, portier 1 à portier 2, portier 2 à point d'extrémité 2, point d'extrémité 1 à portier 2, portier 1 à point d'extrémité 2 ou point d'extrémité 1 à point d'extrémité 2) et recalculé avant l'envoi du message vers le tronçon suivant.

NOTE 3 – L'authentificateur est calculé pour chaque message individuel.

NOTE 4 – Il faut utiliser la méthode de remplissage indiquée dans la norme SHA1 [ISO 10118-3].

NOTE 5 – En cas d'utilisation de l'authentification/intégrité combinées, l'authentificateur est calculé sur l'ensemble du message.

NOTE 6 – Pour éviter le risque d'agression par répétition, il est fortement recommandé de veiller, au niveau de l'implémentation, à changer le mot de passe (clé) avant une rotation complète (soit avant le fin du cycle) du numéro de séquence croissant monotone.

NOTE 7 – Le destinataire a la capacité de détecter l'utilisation de la procédure I par l'évaluation de **algorithmOID** du jeton **EncodedGeneralToken** haché (détectant la présence de "U").

D.6.3.3 Calcul du hachage à mot de passe

L'expéditeur et le destinataire d'un message protégé au niveau de l'authentification/intégrité calculent un hachage dispersé sur clé calculée sur tous les champs de message codés ASN.1 (au moyen de l'identificateur OID "A").

D.6.3.3.1 Code HMAC-SHA1-96

Le code HMAC-SHA1-96 est la valeur hachée cryptographique à 96 bits tronquée de l'algorithme SHA1 à 160 bits. Il faut utiliser comme résultat les 96 bits de poids fort de la représentation de l'ordre des octets du réseau de la valeur de hachage. La référence RFC 2104 décrit la procédure avec la clé secrète *K* à laquelle on attribue la valeur du secret partagé (= mot de passe haché SHA1) et avec *text* auquel on attribue la valeur du "tampon de message".

D.6.3.3.2 Authentification et intégrité

La procédure pour l'authentification et l'intégrité des messages (identificateur OID mis à "A") est la suivante:

l'expéditeur du message doit calculer le hachage de la manière suivante:

1) mettre la valeur de hachage à un modèle par défaut spécifique d'une longueur de 96 bits. La configuration binaire exacte importe peu mais un choix judicieux est une configuration binaire unique qui ne survient pas dans la suite du message;
2) coder l'ensemble des messages en ASN.1;
3) localiser[5] la configuration par défaut dans le message codé; écraser toute la configuration binaire trouvée au moyen de 96 bits zéro;
4) calculer la valeur de hachage cryptographique à partir du message codé en ASN.1 en utilisant le code HMAC-SHA1-96 (voir D.6.3.3.1);
5) substituer, dans le message codé, la configuration par défaut par la valeur de hachage calculée.

Le destinataire recevant le message doit procéder de la manière suivante:

1) décoder le message ASN.1;
2) extraire la valeur de hachage reçue et la conserver dans une RV variable locale;
3) rechercher et localiser la RV de la valeur de hachage dans le message codé reçu;

NOTE – Dans certaines circonstances très rares, la sous-chaîne de la valeur de hachage peut survenir à plusieurs reprises dans l'ensemble du message; il faut, dans ce cas, itérer les étapes 3 à 6 avec des points de départ de la recherche différents.

4) écraser toute la configuration binaire dans le message codé au moyen de 96 zéros;
5) calculer la valeur de hachage cryptographique à partir du message codé au moyen du codage HMAC-SHA1-96 (voir D.6.3.3.1);
6) comparer la RV avec la valeur de hachage calculée. On considère que le message est exempt d'erreur seulement si les deux valeurs de hachage sont égales; dans ce cas, l'authentification est réussie et la procédure s'arrête;
7) sinon, répéter les opérations 3) à 7) en recherchant d'autres concordances après avoir mis la RV à l'emplacement précédent. Si aucune des concordances ne donne une comparaison satisfaisante des valeurs de hachage, l'authentification n'a pas abouti, le message ayant été altéré (accidentellement ou intentionnellement) au cours du transport.

D.6.3.4 Présentation de l'emploi de la Procédure I

Les Figures D.1 à D.3 montrent la présence de clés partagées à l'extrémité des canaux de communication pour les différentes combinaisons de portier et de canaux H.225.0 à routage direct. Indépendamment du modèle d'appel, une clé secrète est toujours présente entre un point d'extrémité et son portier afin de permettre l'authentification/intégrité du message RAS. Lorsqu'un canal RAS et un canal H.225.0 se terminent entre les mêmes (deux) nœuds, on peut utiliser la même clé pour obtenir l'authentification/intégrité des messages RAS et H.225.0.

La Figure D.1 représente le scénario le plus échelonnable dans lequel les deux points d'extrémité sont situés dans des zones qui appliquent le modèle à routage par portier. Tous les portiers concernés partagent mutuellement des clés. On notera que le scénario échelonnable montré à la Figure D.1 ne

[5] Cela peut sous-entendre quelques essais et quelques étapes erronés au cas, très rare, où la configuration par défaut survient plusieurs fois dans le message.

permet pas une sécurité vraie de bout en bout entre des points d'extrémité; tous les aspects sécurité dépendent uniquement des portiers intermédiaires de confiance.

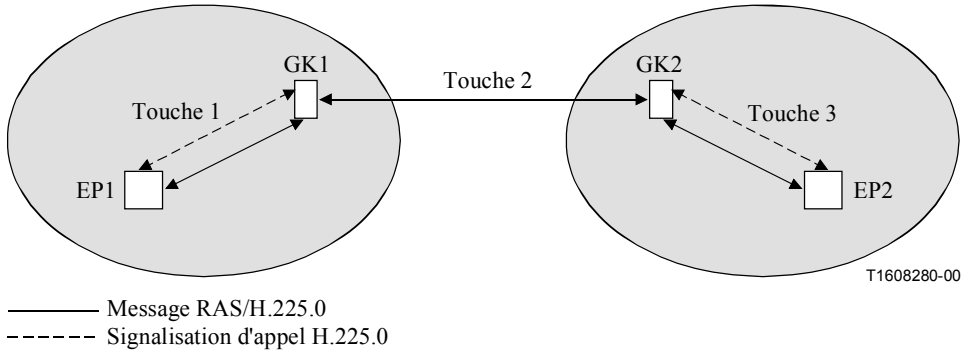


Figure D.1/H.235 – Présentation de l'emploi de la procédure I dans un scénario portier à portier, les deux points d'extrémité se trouvant dans les zones de routage des portiers

La Figure D.2 représente un scénario mixte dans lequel un point d'extrémité se trouve dans une zone appliquant le modèle à routage par portier alors que l'autre point d'extrémité se trouve dans une zone appliquant le modèle à routage direct. Ce scénario peut se produire dans des environnements fermés où le nombre de points d'extrémité 2 et de portiers 1 est restreint.

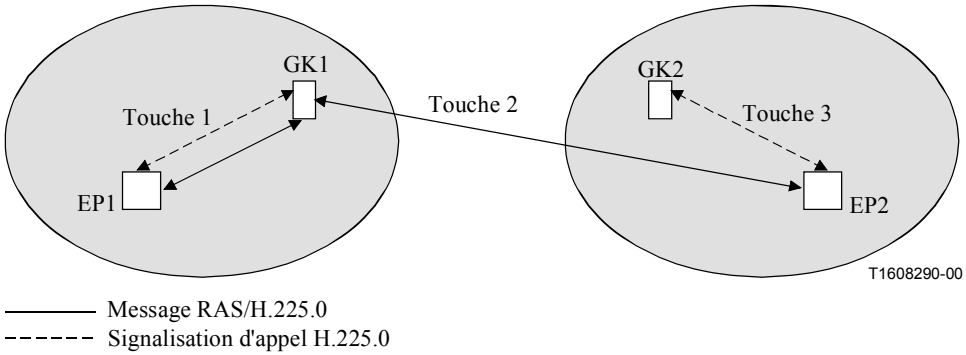


Figure D.2/H.235 – Emploi de la procédure I dans un scénario mixte avec le point d'extrémité 1 dans une zone à routage par portier et le point d'extrémité 2 dans une zone à routage direct

La Figure D.3 représente le scénario dans lequel les deux points d'extrémité se trouvent dans des zones appliquant le modèle du portier à routage direct. Ce scénario n'est pas très échelonnable lorsque plusieurs points d'extrémité sont concernés. En principe, on recommande d'utiliser plutôt l'Annexe E avec les procédures II ou III. Pour ce scénario spécifique et les procédures I, II ou III, il faut des mesures[6] de sécurité additionnelles qui ne sont pas décrites dans la présente Recommandation; elles nécessitent un complément d'étude. On notera que ce scénario assure la sécurité totale de bout en bout sans dépendre des nœuds intermédiaires de confiance.

[6] Protège contre la fraude et le mauvais usage au moyen d'une autorisation d'appel avec jetons d'accès aux passerelles H.323, par exemple.

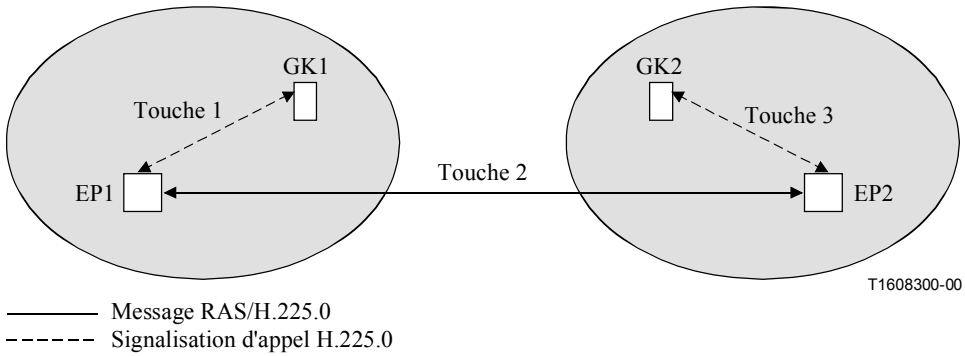


Figure D.3/H.235 – Emploi de la procédure I pour un scénario dans lequel les deux points d'extrémité sont situés dans des zones utilisant un portier à routage direct

Considérons le cas de la Figure D.1 dans lequel trois mots de passe sont partagés par paires entre point d'extrémité 1 et portier 1, entre portier 1 et portier 2 et entre portier 2 et point d'extrémité 2. Trois clés de 20 octets – *Key1*, *Key2* et *Key3* – sont produites à partir de ces mots de passe fondés sur la procédure décrite au 10.3.2. Pour obtenir un maximum de sécurité, il est recommandé de rendre indépendants les trois mots de passe/clés choisis de manière aléatoire.

Les détails des procédures pour l'authentification des messages RAS, H.225.0 et H.245 ainsi que leur intégrité sont présentés ci-après. L'exemple de description illustre les paramètres spécifiques d'un modèle à routage par portier; d'autres combinaisons utiles et valables des identificateurs d'objet dans des scénarios différents sont possibles.

NOTE – Les scénarios présentés dans les Figures 1 à 3 ne se prêtent guère à l'échelonnement lorsque le nombre de clés symétriques (ou mots de passe) utilisés en partage entre les portiers (Figure D.1), entre les portiers et les points d'extrémité distants (Figure D.2) ou entre les points d'extrémité (Figure D.3) est trop élevé.

D.6.3.4.1 Authentification et intégrité des messages RAS

Considérons le cas dans lequel le point EP1 souhaite envoyer un message RAS – un message **ARQ,** par exemple – au portier GK1. Le point EP1 produit un horodateur et un numéro de séquence et les introduit dans les champs **timeStamp** et **random** respectivement, avec l'alias du portier GK1 dans **generalID** et l'identificateur du point EP dans le champ **sendersID**. Ces champs apparaissent dans le champ **ClearToken** de **hashedVals**, lui-même faisant partie de **cryptoHashedToken** du champ **CryptoToken** du **cryptoH323Token** du message **ARQ**.

Le champ **tokenOID** de **cryptoHashedToken** est mis à "A", ce qui indique que tous les champs du message **ARQ** sont hachés. Le champ **HASHED** dans **token** de **cryptoHashedToken** a le champ **algorithmOID** mis à "U", ce qui indique l'utilisation du code HMAC-SHA1-96 et le champ **params** mis à NULL. Le point EP1 calcule ensuite l'authentificateur sur la base du code HMAC-SHA1-96 en utilisant la clé *Key1* à 12 octets. L'authentificateur est calculé sur l'ensemble du message RAS.

Le point EP1 introduit un authentificateur calculé dans le champ **hash** de **token** du champ **cryptoHashedToken** de **CryptoToken** qui est présent dans le champ **cryptoH323Token** du message **ARQ**. Celui-ci est ensuite envoyé au portier GK1.

Lorsqu'il reçoit le message **ARQ**, le portier GK1 vérifie l'authentificateur sur la base de divers critères qui englobent:

- l'actualité de **timestamp** et l'unicité de **random**;
- l'identité de **generalID** et de son propre identificateur;
- la concordance de l'authentificateur du message **ARQ** et de l'authentificateur calculé par le portier GK1.

D.6.3.4.2 Authentification et intégrité du message H.225.0

Considérons le cas dans lequel le point d'extrémité EP1 souhaite envoyer au point d'extrémité EP2 un message H.225.0, un message **Setup** par exemple. Le point EP1 produit un horodateur et un numéro de séquence et les introduit dans les champs **timeStamp** et **random** respectivement, avec l'alias du portier GK1 dans **generalID** et l'identificateur du point EP dans le champ **sendersID**. Le point EP1 calcule également une demi-clé Diffie-Hellman et introduit les paramètres Diffie-Hellman **halfkey**, **modsize** et **generator** dans le champ **dhkey** de **ClearToken**. Ces champs se trouvent dans le champ **ClearToken** de **hashedVals**, lui-même se trouvant dans **cryptoHashedToken** du champ **CryptoToken** de **cryptoH323Token** du message **Setup**.

Le champ **tokenOID** de **cryptoHashedToken** est mis à "A" pour indiquer que tous les champs du message **Setup** sont hachés. Le champ **HASHED** de **token** dans **cryptoHashedToken** a son champ **algorithmOID** mis à "U" pour indiquer l'utilisation du code HMAC-SHA1-96 et **params** mis à NULL. Le point EP1 calcule ensuite l'authentificateur sur la base du code HMAC-SHA1 au moyen de la clé *Key1* à 12 octets. L'authentificateur est calculé conformément à la méthode de hachage choisie (A) et compte tenu de l'ensemble du message H.225.0.

Le point EP1 introduit l'authentificateur calculé dans **hash** du champ **token** du champ **cryptoHashedToken** de **CryptoToken,** présent dans **cryptoH323Token** du message **Setup**. Le message **Setup** est ensuite envoyé au portier GK1.

Lorsqu'il reçoit le message **Setup**, le portier GK1 vérifie l'authentificateur sur la base de divers critères qui englobent:

- l'actualité de **timestamp** et l'unicité de **random**;
- l'identité de **generalID** et de son propre identificateur;
- la vérification des paramètres Diffie-Hellman, par exemple en contrôlant si les paramètres prime et generator à 1024 bits sont corrects. La vérification de la sûreté des paramètres DH est une opération longue qui n'aura lieu que si la politique locale l'exige;
- la concordance de l'authentificateur du message **Setup** et de l'authentificateur calculé par le portier GK1.

Si la vérification de l'authentificateur est positive, le portier GK1 calcule un nouvel authentificateur qu'il substituera à l'ancien dans le message **Setup** avant de l'envoyer au portier GK2 de la manière qui suit: le portier GK1 remplace les valeurs de **timeStamp**, **random**, **sendersID** et **generalID** du champ **ClearToken** de **hashedVals** par des valeurs qui s'appliquent au tronçon GK1-GK2. Le champ **timestamp** contient l'horodateur en vigueur, le champ **random** contient le numéro séquentiel croissant monotone du tronçon GK1-GK2, le champ **generalID** contient l'alias du portier GK2 et **sendersID** contient l'alias du portier GK1. Celui-ci introduit également les paramètres Diffie-Hellman reçus dans le champ **dhkey** de **ClearToken**.

Le portier GK1 calcul ensuite le nouvel authentificateur pour ce message **Setup** au moyen de la clé *Key2* et de l'algorithme HMAC-SHA1-96 (**algorithmOID**="U"), l'introduit dans le champ **hash** de **token** et transmet le message **Setup** au portier GK2.

Lorsqu'il reçoit le message **Setup**, le portier GK2 vérifie l'authentificateur, calcule un nouvel authentificateur après avoir modifié les champs **ClearToken** de **hashedVals** de manière appropriée, l'introduit dans le champ **hash** et transmet le message **Setup** au point d'extrémité EP2.

D.6.3.4.3 Authentification et intégrité de message H.245

Considérons le cas dans lequel le point EP1 souhaite envoyer un message H.245 – un message **TerminalCapabilitySet** par exemple – au point EP2. Le point EP1 vérifie s'il y a lieu d'envoyer un message H.225.0 au portier GK1. Si c'est le cas, le message H.245 est canalisé en tunnel dans ce message H.225.0. Les champs contenus dans le message H.225.0 ont les valeurs indiquées précédemment pour la transmission d'un message H.225.0. Étant donné que le message H.245 est

canalisé en tunnel, le champ **h323-uu-pdu** du message **h323-UserInformation** prend les valeurs suivantes:

- le champ **h323-message-body** est mis au type de message H.225.0 en cours de transmission;
- **h245Tunnelling** est mis à TRUE;
- **h245Control** contient la chaîne d'octets PDU H.245.

Le point EP1 produit un **CryptoToken** pour le message H.225.0, met **tokenOID** à "A" pour indiquer l'authentification et l'intégrité, met **timeStamp**, **random**, **sendersID**, **generalID** et **tokenOID** à "T" dans **ClearToken** de **hashedVals**, met **algorithmOID** à "U" pour indiquer l'utilisation du code HMAC-SHA1-96 et **hash** à l'authentificateur de hachage calculé sur l'ensemble des champs du message **H323-UU-PDU**.

Toutefois, si aucune transmission de message H.225.0 est en attente, le message H.245 sera canalisé en tunnel dans un message **facility** H.225.0 ad hoc. Le champ **h323-uu-pdu** du message **h323-UserInformation** contient les valeurs suivantes:

- **h323-message-body** est mis à **facility** qui contient:
  - **reason** mis à **undefinedReason**;
  - **tokens** et **cryptoTokens** comme pour n'importe quel message H.225.0;
- **h245Tunnelling** mis à TRUE;
- **h245Control** contient la chaîne d'octets PDU H.245.

Comme indiqué ci-dessus, le point EP1 produit un **CryptoToken** dans le cadre du message **facility** H.225.0. Le message **facility** est ensuite transmis par le point EP1 au portier GK1.

Dans les deux cas (la transmission d'un message H.225.0 en attente ou l'utilisation d'un message **facility** H.225.0 ad hoc), le portier GK1 vérifie l'authentificateur à la réception du message. Ensuite, si une transmission de message H.225.0 est en attente pour le tronçon GK1-GK2, le message H.245 est canalisé en tunnel dans ce message; sinon, il est canalisé en tunnel dans un message **facility** H.225.0 ad hoc. Comme pour toutes les transmissions de message H.225.0, un nouvel authentificateur est calculé pour le message en question avant sa transmission du portier GK1 au portier GK2. Le processus se répète pour le tronçon GK2-EP2.

D.6.4 Scénario de routage direct

Les entités H.323 sécurisées peuvent communiquer non seulement dans le contexte de routage par portier comme indiqué dans la présente Recommandation mais peuvent également appliquer le modèle à routage direct. Celui-ci nécessite des mesures de sécurité additionnelles (jetons d'accès) qui ne sont pas nécessaires dans les environnements plus simples à routage par portier. La sécurisation du modèle à routage direct nécessite donc un complément d'étude.

D.6.5 Prise en charge du service de réalisation d'extrémité

Les entités H.323 sécurisées peuvent utiliser les services de réalisation d'extrémité conformément à la procédure décrite dans l'Appendice I.4.6.

D.6.6 Compatibilité avec le contexte H.235 Version 1

Bien que ces profils de sécurité soient mis au point dans le contexte H.235 version 2 [H.235 (2000)], il est possible de les appliquer dans un contexte H.235 version 1 [H.235 (1998)] moyennant quelques modifications mineures. Un destinataire a la capacité de détecter la présence de la version du protocole H.235 de l'expéditeur par l'évaluation des identificateurs d'objet du profil de sécurité (voir D.11).

Réalisations H.235 version 1 [H.235 (1998)]:

- ne pas attribuer de valeur à ou ne pas évaluer **sendersID** de **ClearToken**;
- ne pas utiliser les services d'extrémité comme indiqué au D.6.5.

D.6.7 Comportement en multidiffusion

Les messages multidiffusés H.225.0 tels que les messages GRQ ou LRQ ne doivent pas comporter le CryptoToken requis par la procédure I, sauf lorsqu'ils sont envoyés en monodiffusion.

D.7 Profil de sécurité de cryptage vocal

La procédure générale établit un secret partagé (échange Diffie-Hellman) entre les deux parties en communication au lancement de la connexion. Ce secret partagé est ensuite utilisé pour protéger (un ensemble de) des touches média qui sont utilisées pour crypter les sessions médias (RTP).

Le profil de sécurité de cryptage vocal est une amélioration facultative du profil de sécurité élémentaire et du profil de sécurité de signature; son utilisation peut être négociée dans le contexte de la négociation de la capacité de sécurité du terminal. Dans les environnements où la confidentialité vocale est assurée par d'autres moyens, il n'est pas nécessaire de mettre en œuvre le cryptage du média et les procédures de gestion de clés correspondantes (concordance de clés Diffie-Hellman, mise à jour et synchronisation de clés).

Les algorithmes de cryptage choisis sont: compatible RC2, DES et DES triple. On notera que la norme DES triple pouvant être utilisée par l'algorithme DES, cela permet d'obtenir une réalisation compacte. Indépendamment du choix de l'algorithme de cryptage de média spécifique, les options ci-après doivent être suivies explicitement:

- si nécessaire, génération d'un vecteur d'initialisation (IV), spécifié au B.3 a);
- si nécessaire, remplissage comme indiqué dans l'Annexe B.

La capacité utile[7] audio est cryptée au moyen de l'algorithme de cryptage négocié ("X", "Y" ou "Z") fonctionnant en mode CBC conformément aux procédures décrites au paragraphe 11 et dans l'Annexe B et aux méthodes de bourrage cryptographiques de l'Appendice I.1.

D.7.1 Gestion de clés

- Au cours de la séquence **Setup** à **Connect** a lieu un échange Diffie-Hellman (DH), qui confère aux deux points d'extrémité un secret partagé. Le champ **ClearToken** des champs **CryptoToken** doit contenir une clé **dhkey**, utilisée pour transmettre les paramètres comme indiqué dans la présente Recommandation. Le champ **halfkey** contient la clé publique aléatoire d'une partie, **modsize** contient DH-prime et **generator** contient DH-group. Les paramètres DH à utiliser sont indiqués dans le Tableau 4. Pour plus de détails, se référer à [RFC 2412, Appendice E2]. On notera que les messages H.225.0 étant authentifiés (comme décrit précédemment dans la procédure I), l'échange DH est un échange authentifié.
- Au cours de la connexion rapide (FastStart), le demandeur (source de **Setup**) présente à la fois son jeton DH et les structures FastStart qu'il prend en charge. Les deux canaux H235Cap et nonH235Cap devraient être proposés. Le champ **mediaWaitForConnect** doit être mis à Vrai[8].

---

[7] Sans en-tête de capacité utile.

[8] On notera que dans ce cas, si l'appelé transmet le média crypté à l'appelant (ce qu'il peut faire théoriquement étant donné qu'il dispose des adresses RTP/RTCP de l'appelant), l'appelant ne sera pas en mesure de le déchiffrer sans le secret partagé fourni dans le message Connect (Alerting, Call Proceeding). (Pour les besoins de la relation de sécurité, l'appelé est a priori le maître.)

- Au cours de la connexion rapide FastStart, l'appelé (source de **Connect**) présente à la fois son jeton DH et les structures FastStart qu'il accepte. La clé de session est indiquée dans le champ **encryptionSync**. La clé de session est elle-même cryptée au moyen du secret partagé DH comme dans une connexion normale (non-FastStart).
- Au cours de l'échange de capacités H.245, les points d'extrémité présentent des entrées **H235Capability** pour les codes qu'ils prennent en charge. Chaque codec est associé à une capacité H.235 individuelle. Ces capacités devraient indiquer la prise en charge de la norme compatible RC2 à 56 bits (OID – "X"), de la norme DES à 56 bits (OID – "Y") ou de la norme DES triple à 168 bits (OID – "Z").

  Les algorithmes de cryptage négociés et leurs modes de fonctionnement pour le cryptage du flux média seront également utilisés pour la distribution sécurisée de la clé de session. L'algorithme de cryptage utilisé pour le cryptage de média doit fonctionner dans le même mode de chaînage que l'algorithme de cryptage de média.
- Les réponses **OpenLogicalChannel(Ack)** sont émises avec la clé de session (principale) créée et introduite dans le champ **encryptionSync**. La clé de session est elle-même cryptée au moyen du secret partagé DH de la manière décrite ci-dessous[9].
- **OpenLogicalChannel** achemine à la fois **forwardLogicalChannelParameters** et **reverseLogicalChannelParameters** avec **dataType** fournissant **h235Media** avec **encryptionAuthenticationAndIntegrity** où un **MediaEncryptionAlgorithm** au maximum doit être présent dans **encryptionCapability**.

NOTE – S'il n'y a pas d'algorithme de cryptage disponible des deux côtés, le flux média doit être laissé non crypté ou la connexion peut être arrêtée prématurément, selon la politique de sécurité.

- La clé de session cryptée doit être acheminée dans le champ H.235Key/**sharedSecret** du champ **encryptionSync**. La clé de session doit être acheminée dans le champ **keyMaterial** de **KeySyncMaterial**. Celui-ci est crypté au moyen:
  - des 56 bits du secret partagé, à commencer par les bits de faible poids du secret Diffie-Hellman pour l'identificateur OID "X" ou OID "Y";
  - de tous les bits du secret partagé pour l'identificateur OID "Z" commençant par les bits de faible de poids du secret DH:

Il convient d'inclure la valeur **generalID** pour fournir un niveau minimum d'authentification de la source de la clé de session (voir également D.7.2). Le destinataire devrait vérifier l'exactitude de la valeur **generalID** reçue.

Chaque entité doit prendre les bits de faible poids appropriés provenant du secret Diffie-Hellman partagé commun pour la clé de cryptage principale (clé principale); c'est-à-dire les 56 bits de plus faible poids du secret Diffie-Hellman pour l'identificateur OID "X" ou OID "Y" et les 168 bits de plus faible poids provenant du secret Diffie-Hellman pour l'identificateur OID "Z".

---

[9] On notera qu'il n'y a pas de méthode prescrite pour produire les clés de session qui sont utilisées pour crypter le média. La production de ces valeurs est une question d'implémentation qui dépend des ressources, de la politique et de l'algorithme de cryptage à utiliser. Il convient de prendre garde d'éviter de produire des clés faibles.

Tableau D.4/H.235 – Groupes Diffie-Hellman

OID	Description de groupe D-H
"X" (compatible RC2), "Y" (DES)	Mod-P, tous les 512 bits premiers appropriés
"Z" (triple-DES)	Mod-P, 1024 bits primaires Primaire = $2^{1024} – 2^{960} – 1 + 2^{64} × \{ [2^{894}$ pi] + 129093 } = (179769313486231590770839156793787453197860296048756011706444 423684197180216158519368947833795864925541502180565485980503 646440548199239100050792877003355816639229553136239076508735 759914822574862575007425302077447712589550957937778424442426 617334727629299387668709205606050270810842907692932019128194 467627007)$_{10}$ Générateur[a] = 2
[a] Le générateur est utilisé pour produire le jeton DH.	

D.7.2 Mise à jour et synchronisation des clés

Le taux de rafraîchissement des clés *doit* être tel que pas plus de 2^{32} blocs soient cryptés au moyen de la même clé. Il *convient* que les implémentations rafraîchissent les clés avant que 2^{30} blocs aient été cryptés au moyen de la même clé (voir 10.3). Les deux entités concernées sont libres de changer la clé de session média aussi souvent qu'elles le jugent nécessaire compte tenu de leur politique de sécurité. Par exemple, le maître peut distribuer une nouvelle clé de session au moyen de **encryptionUpdate** du message **miscellaneousCommand**. Par ailleurs, l'esclave peut demander au maître une nouvelle clé de session afin de la changer au moyen de **encryptionUpdateRequest** du message **miscellaneousCommand**.

Le message **MiscellaneousCommand** contient **encryptionUpdate** dont le champ **encryptionSynch** est mis aux paramètres suivants:

- **synchFlag**: le nouveau numéro de capacité utile RTP dynamique indiquant un changement de clé.
- **h235key**: achemine la nouvelle clé de session cryptée. Il s'agit de la clé **H235Key** codée en ASN.1 transmise comme une chaîne d'octets.

Le champs **sharedSecret** dans la structure **H235Key** utilise les champs suivants:

- **algorithmOID**: mis à "X" pour la norme compatible RC2 à 56 bits, à "Y" pour la norme DES à 56 bits ou "Z" pour la norme DES triple à 168 bits. Il s'agit d'un algorithme de cryptage par lequel la clé de session média est en cours de cryptage.

NOTE 1 – L'algorithme de cryptage de clé de session est le même que l'algorithme de cryptage média négocié.

- **paramS**: mis à la valeur initiale. Le champs **iv8** contient une configuration binaire formée d'un bloc de 64 bits aléatoires produit par l'initiateur. Ce champ n'est pas utilisé pour le mode CBC et sa valeur est NULL.
- **encryptedData:** mis au résultat de **KeySynchMaterial** crypté.

En tant que partie de **KeySyncMaterial**:

- **generalID**: identificateur de la source distribuant la clé.

- **keyMaterial**: mis à la nouvelle clé de session. Pour les normes DES et compatible RC2, il s'agit d'une clé à 56 bits, pour la norme DES triple, une clé de 168 bits. Le maître doit produire une nouvelle clé de session qui répond au moins aux critères de sécurité suivants: il ne s'agira pas d'une clé DES faible ou semi-faible et elle doit utiliser une source aléatoire suffisamment sûre.

Le message **MiscellaneousCommand** contient le champ **encryptionUpdateRequest** contenant **keyProtectionMethod** où le fanion **sharedSecret** est mis à TRUE.

NOTE 2 – Etant donné que la mise à jour et la synchronisation de la clé dépendent des messages H.245 qui ne sont pas superposés au cours de la connexion rapide, il faut utiliser une canalisation en tunnel H.245 pour les entités H.323 sécurisées. Ainsi, la mise à jour et la synchronisation des clés ne peuvent être utilisées que dans le profil de sécurité de signature.

D.7.3 Normes DES triples en mode CBC extérieur

Il *convient* d'utiliser, dans le présent profil de sécurité, la norme DES triple à 168 bits en mode CBC extérieur, présenté dans la Figure D.4. Dans cette figure, chaque indice k_i se rapporte à une clé à 56 bits. Il *faut* utiliser une clé à 56 bits différente dans chaque bloc de cryptage (E) et de décryptage (D). Aucune des clés faibles à 64 bits de la norme DES n'est réputée entraîner une faiblesse dans la norme DES triple. Néanmoins, les implémentations qui sont conformes à ce profil devraient refuser la clé s'il s'agit d'une clé DES [voir RFC 2405].

De plus amples informations sur la norme DES triple sont données dans [Schneier] et [RFC 2405].

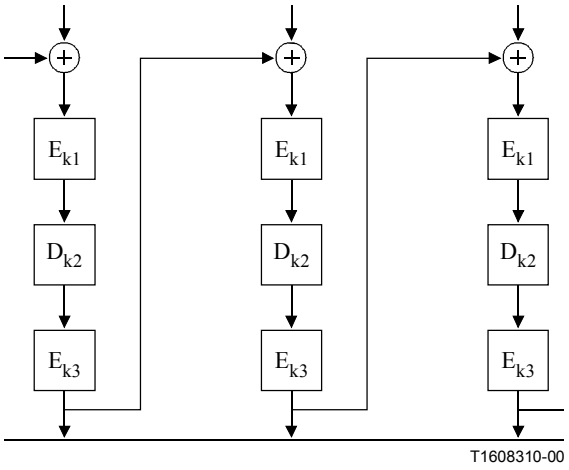


Figure D.4/H.235 – Cryptage DES triple en mode CBC extérieur

D.8 Interception licite

A étudier (voir [LI]).

D.9 Liste des messages de signalisation sécurisés

Le présent paragraphe explique brièvement la manière et les moyens de l'Annexe D pour sécuriser les divers messages de signalisation H.323.

D.9.1 Message RAS H.225.0

Message RAS H.225.0	Champs de signalisation H.235	Authentification et intégrité
Tous	cryptoTokens	Procédure I

D.9.2 Signalisation d'appel H.225.0

Message de signalisation H.225.0	Champs de signalisation H.235	Authentification et intégrité
Alerting-UUIE, CallProceeding-UUIE, Connect-UUIE, Setup-UUIE, Facility-UUIE, Progress-UUIE, Information-UUIE, ReleaseComplete-UUIE	cryptoTokens	Procédure I

D.9.3 Commande d'appel H.245

Les messages H.245 provenant de ou destinés à des entités H.323 sécurisées doivent être superposés dans le cadre de la connexion rapide et sécurisée ou doivent être canalisés en tunnel dans un message **Facility-UUIE** H.225.0 sécurisé.

D.10 Utilisation des identificateurs sendersID et generalID

Le paramètre **ClearToken** contient les champs d'identificateurs **sendersID** et **generalID**. Lorsque l'information d'identification est disponible, la valeur de l'identificateur **sendersID** est celle de l'identificateur du portier (GKID, *gatekeeper identifier*) pour le message provenant du portier, et celle de l'identificateur du point d'extrémité (EPID, *endpoint identifier*) pour les messages provenant du point d'extrémité. Lorsque l'information d'identification est disponible, la valeur de l'identificateur **generalID** est celle de l'identificateur du portier (GKID) pour les messages provenant du point d'extrémité, et celle de l'identificateur du point d'extrémité (EPID) pour les messages provenant du portier. Lorsque l'information d'identification n'est pas disponible ou lorsque la diffusion/multidiffusion est ambiguë, le champ est absent ou contient une chaîne de zéros. Le Tableau D.5 résume la situation:

Tableau D.5/H.235 – Identificateurs d'objet utilisés dans l'Annexe D

Message	sendersID	generalID
Unicast **GRQ**	**EPID** si disponible, autrement **NULL**	**GKID**
Multicast **GRQ**	**EPID** si disponible, autrement **NULL**	
GCF, **GRJ**	**GKID**	**EPID** si disponible, autrement **NULL**
Initial **RRQ**	**EPID** si disponible, autrement **NULL**	**GKID**
RCF	**GKID**	**EPID**
RRJ	**GKID**	

Tableau D.5/H.235 – Identificateurs d'objet utilisés dans l'Annexe D (suite)

Message	sendersID	generalID
URQ, **UCF**, **URJ**, **BRQ**, **BCF**, **BRJ**, **DRQ**, **DCF**, **DRJ**, **NSM**, **RIP**, **SCI**, **SCR**, **XRS** (EP-to-GK)	**EPID**	**GKID**
URQ, **UCF**, **URJ**, **BRQ**, **BCF**, **BRJ**, **DRQ**, **DCF**, **DRJ**, **NSM**, **RIP**, **SCI**, **SCR**, **XRS** (GK-to-EP)	**GKID**	**EPID**
ARQ, **IRQ**, **RAI**	**EPID**	**GKID**
ACF, **ARJ**, **BCF**, **LCF**, **LRJ**, **IRR**, **IRQ**, **RAC**, **LCF**, **LRJ**, **IACK**, **INAK**	**GKID**	**EPID**
Unicast **LRQ** (EP-to-GK)	**EPID**	**GKID**
Unicast **LRQ** (GK-to-GK)	**GKID**	**GKID**
Multicast **LRQ**	**EPID**	
NOTE – GKID désigne l'identificateur du portier, EPID désigne l'identificateur du point d'extrémité. L'espace vide indique une chaîne d'identification manquante ou nulle.		

D.11 Liste d'identificateurs d'objet

Le Tableau D.6 ci-dessous énumère tous les identificateurs OID mentionnés (voir également [OIW] et [WEBOID]). Il y a des identificateurs d'objet pour H.235v1 [H.235 (1998)] et pour H.235v2 [H.235 (2000)].

Tableau D.6/H.235 – Identificateurs d'objet utilisés dans l'Annexe D

Désignation de l'identificateur d'objet	Valeur(s) de l'identificateur d'objet	Description
"A"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 1} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 1}	Utilisé dans la procédure I pour l'indicateur CryptoToken-tokenOID indiquant que le hachage englobe tous les champs du message RAS/H.225.0 (authentification et intégrité)
"T"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 5} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 5}	Utilisé dans la procédure I pour l'identificateur ClearToken-tokenOID indiquant que ClearToken est en cours d'utilisation pour l'authentification et l'intégrité des messages.
"U"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 6} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 6}	Utilisé dans la procédure I pour l'identificateur Algorithm OID indiquant l'utilisation du code HMAC-SHAI-96.
"X"	{iso(1) member-body(2) us(840) rsadsi(113549) encryptionalgorithm(3) 2}	Cryptage vocal utilisant un algorithme compatible RC2 (56 bits) ou compatible RC2 en mode CBC et groupe DH à 512 bits.
"Y"	{iso(1), identified-organization(3), oiw(14), secsig(3), algorithm(2), descbc(7)}	Cryptage vocal utilisant l'algorithme DES (56 bits) en mode CBC et le groupe DH à 512 bits.
"Z"	{iso(1) identified-organization(3) oiw(14) secsig(3) algorithm(2) desEDE(17)}	Cryptage vocal utilisant l'algorithme DES triple (168 bits) en mode CBC extérieur et groupe DH à 1024 bits.

D.12 Références bibliographiques

[FIPS PUB 180-1] NIST, FIPS PUB 180-1: Secure Hash Standard, Avril 1995 http://csrc.nist.gov/fips/fip180-1.ps

[LI] Draft DRT/TIPHON-08003 V0.0.9, "Lawful Interception – Internal LI Interface", August 2000.

[OIW] Stable Implementation – Agreements for Open Systems Interconnection Protocols: Part 12 – OS Security; Output from the December 1994 Open Systems Environment Implementors' Workshop (OIW); http://nemo.ncsl.nist.gov/oiw/agreements/stable/OSI/12s_9412.txt

[RFC 2405] C. Madson, N. Doraswamy "The ESP DES-CBC Cipher Algorithm With Explicit IV", RFC 2405, *Internet Engineering Task Force,* 1998

[WEBOIDs] http://www.alvestrand.no/objectid/top.html

ANNEXE E

Profil de signature

E.1 Aperçu général

La présente annexe décrit un profil de sécurité à signatures numériques qui est proposé à titre d'option. Les entités de sécurité H.323 (terminaux, portiers, passerelles, ponts MCU, etc.) peuvent implémenter ce profil de sécurité de signature pour améliorer la sécurité ou pour l'assurer en cas de nécessité.

Le profil de sécurité de signature autorise le modèle à routage par portier; il est basé sur les techniques de canalisation en tunnel H.245. La prise en charge de modèles autres que ceux routés par portier nécessite un complément d'étude.

Le profil de sécurité de signature est applicable à la téléphonie IP "globale" échelonnable; ce profil de sécurité n'est pas exposé aux limitations du profil de sécurité élémentaire simple de l'Annexe D. Par exemple, le profil de sécurité de signature ne dépend pas de l'administration des secrets partagés mutuels, des bonds dans différents domaines. Il assure la canalisation en tunnel des messages H.245 pour l'intégrité de ceux-ci et offre également des dispositions pour la non-répudiation des messages. Le profil de sécurité de signature offre ainsi la sécurité bond par bond ainsi que l'authentification vraie de bout en bout avec l'utilisation simultanée de serveurs mandataires H.235 ou de portiers intermédiaires.

Les caractéristiques proposées par ces profils sont: pour les messages RAS, H.225.0 et H.245:

- authentification de l'utilisateur jusqu'à une entité voulue indépendamment du nombre de bonds[10] au niveau application qu'effectue le message;
- intégrité de toutes les parties déterminantes (champs) des messages arrivant à une entité, indépendamment du nombre de bond au niveau application qu'effectue le message. L'intégrité du message proprement dite obtenue au moyen d'un nombre aléatoire fort est proposée en option;
- l'authentification du message bond par bond au niveau application, l'intégrité et la non-répudiation assurent ces services de sécurité pour l'ensemble du message;
- la non-répudiation des messages échangés entre deux entités, indépendamment du nombre de bond au niveau application qu'effectue le message, peut également être assurée. Plus précisément, la non-répudiation est assurée pour des parties déterminantes (champs) du message. Cela sera par exemple le cas lorsqu'un point d'extrémité EP envoie un message SETUP à son portier et que ceux-ci (le point EP et le portier) sont séparés par un ou plusieurs serveurs mandataires.

La résistance à diverses agressions est correctement assurée au moyen des services de sécurité ci-dessous, à savoir:

- agressions visant la fonction de refus de service: un contrôle rapide des signatures numériques peut prévenir de telles agressions;
- agressions par entremetteur: l'authentification et l'intégrité du message bond par bond au niveau application protègent contre de telles agressions lorsque l'entremetteur se trouve entre une bond au niveau application et un routeur hostile, par exemple. Lorsque l'entremetteur est une entité de niveau application, de telles agressions sont empêchées par la présence de

[10] Par "bond", on entend dans le cas présent un élément de réseau H.235 de confiance (tel que portier, passerelle, pont MCU, serveur mandataire ou pare-feu). Donc, la sécurité bond par bond de niveau application, lorsqu'elle est utilisée avec des techniques symétriques, ne donne pas une sécurité vraie de bout en bout entre les terminaux.

l'authentification et de l'intégrité de l'utilisateur de bout en bout pour des parties sélectionnées du message;

- agressions par répétition: l'emploi d'horodateurs et de numéros de séquence protège contre de telles agressions;
- parodie: l'authentification de l'utilisateur protège contre de telles agressions;
- détournement de la connexion: l'utilisation de l'authentification/intégrité pour chaque message de signalisation empêche de telles agressions.

E.2 Conventions de spécification

Le profil de sécurité de signature peut utiliser le **profil de sécurité de cryptage vocal** de l'Annexe D pour réaliser la confidentialité vocale si nécessaire.

Les procédures II et III spécifient la manière d'implémenter les services de sécurité pour divers scénarios – bond par bond et de bout en bout – avec des mécanismes de sécurité différents tels que les techniques de cryptographie asymétrique (signature numérique).

Si le service d'intégrité des messages fournit en outre toujours l'authentification des messages, l'inverse n'est pas toujours vrai. En mode d'authentification seule, l'intégrité assurée porte uniquement sur un sous-ensemble donné de champs de messages. Cela s'applique aux services d'intégrité assurés par des moyens asymétriques (par exemple les signatures numériques). Donc, en pratique, le service d'authentification et d'intégrité combinées exploite les mêmes données relatives aux clés sans introduire de faiblesse au niveau de la sécurité.

Par ailleurs, toutes les informations de sécurité bond par bond sont introduites dans l'élément **CryptoSignedToken**. Ces informations sont recalculées à chaque bond, conformément à la procédure II.

Par ailleurs, les informations de sécurité de bout en bout – uniquement possibles en cas d'utilisation d'un serveur mandataire H.323 et de la procédure III – calculent essentiellement les informations analogues à celles placées dans **CryptoSignedToken** mais les enregistrent dans un jeton **CryptoToken** indépendant du message. Ces informations ne sont pas modifiées pendant le transport. Un identificateur d'objets séparé permet de faire la distinction entre les jetons **CryptoToken** bond par bond et de bout en bout.

Autorité de certification: lorsqu'elles sont utilisées dans le contexte de la signature électronique, les Autorités de certification (CA) certifient les clés de vérification publique par l'émission de "Certificats".

Dépôt de certificat: les dépôts de certificat (par exemple un répertoire X.500) contiennent des certificats d'utilisateur et des listes d'annulation de certificats (CRL). Ils bénéficient de la confiance lorsqu'ils donnent accès à ces informations accessibles mais ne sont pas responsables du contenu ou de l'exactitude des informations qu'ils reçoivent des entités CA et RA.

Signature numérique: transformation cryptographique (au moyen d'une technique cryptographique asymétrique) de la représentation numérique d'un message de données de telle manière que toute personne ayant le message signé et la clé publique qui s'applique peut déterminer:

i) si la transformation a été faite au moyen de la clé privée correspondant à la clé publique en question;

ii) si le message signé n'a pas été altéré depuis la transformation cryptographique.

Fournisseurs en ligne de statut de certificat: le protocole de statut de certificat en ligne (OCSP) permet à des applications de déterminer l'état d'annulation d'un certificat identifié. Le protocole OCSP peut être utilisé pour satisfaire certaines des conditions opérationnelles de fourniture d'informations d'annulation plus rapidement que cela n'est possible avec les listes CRL. On peut

considérer les fournisseurs de statut de certificat en ligne comme une alternative à l'emploi des listes CRL hors ligne.

Serveur mandataire: le serveur mandataire est une entité H.323 intermédiaire analogue à un portier. Il peut être un nœud de réseau séparé ou peut être situé au même endroit que la fonctionnalité d'une entité H.323, par exemple celle d'un portier. Le serveur mandataire peut effectuer des tâches de sécurité telles que la vérification de signatures et de certificats ainsi que le contrôle d'accès.

Autorité d'enregistrement: les autorités d'enregistrement agissent comme des intermédiaires entre les utilisateurs et les CA. Elles reçoivent des demandes émanant des utilisateurs et les transmettent aux CA dans un formulaire approprié.

Autorités d'horodatage: les autorités d'horodatage sont obligatoires pour la non-répudiation en cas de perte de clé ou de compromis au sujet d'une clé. Dans la pratique, elles fournissent à quiconque une contre signature, avec une heure fiable, via un hachage et un identificateur de hachage.

Fournisseur de service de confiance: entité qui peut être utilisée par d'autres entités comme intermédiaires de confiance dans un processus de communication ou de vérification, ou comme un fournisseur de services d'information de confiance.

Le profil de sécurité est proposé en option. Il est applicable dans les environnements pouvant avoir de nombreux terminaux et où l'attribution des mots de passe/clés symétriques n'est pas réalisable, par exemple dans un scénario à grande échelle ou à l'échelle mondiale. Le profil de sécurité de signature fournit des services de sécurité additionnels pour la non-répudiation en utilisant des signatures et des certificats numériques. Les signatures numériques pourraient utiliser le hachage SHA1 ou MD5 et fournir l'authentification et/ou l'intégrité (voir les procédures II et III).

Les entités H.323 utilisant l'authentification et l'intégrité ou l'authentification seule au niveau bond par bond doivent utiliser la procédure II. Les entités H.323 utilisant l'authentification seule ne mettraient pas en œuvre l'intégrité. Les entités H.323 utilisant l'authentification seule doivent utiliser la procédure III pour réaliser l'authentification vraie de bout en bout.

Le profil de sécurité de signature permet de canaliser en tunnel, en toute sécurité, les ponts PDU de commande dans des messages de fonction H.225.0. Les mécanismes de mise à jour et de synchronisation des clés H.245 nécessitent une canalisation en tunnel qui est utile, par exemple, dans le cas de communications[11] de très longue durée.

La zone hachurée verticalement – en jaune dans la version électronique du Tableau E.1 représente le domaine du profil de sécurité de signature. Lorsqu'on omet l'intégrité, signalée dans la zone hachurée verticalement – en orange dans la version électronique, on obtient le profil de sécurité d'authentification seule. Une option dans le profil de sécurité de signature consiste à faire un choix entre les signatures numériques RSA-SHA-1 et RSA-MD5. Le profil de sécurité de cryptage vocal de l'Annexe D (voir D.7) pourrait être facultativement utilisé en combinaison avec le profil de sécurité de signature.

[11] La mise à jour des clés pour le codage de la parole G.711 sécurisé devrait intervenir après la transmission de 2^{30} blocs de 64 bits, soit plus de 12 jours de conversation continue.

Tableau E.1/H.235 – Profil de sécurité de signature

Services de sécurité	Fonctions d'appel			
	RAS	H.225.0	H.245[a)]	RTP
Authentification	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	
Non-Répudiation	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	
Intégrité	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	SHA1/ MD5 signature numérique	
Confidentialité				
Contrôle d'accès				
Gestion de clés	attribution d'un certificat	attribution d'un certificat		

a) H.245 canalisé en tunnel ou H.245 incorporé dans connexion rapide H.225.0.

NOTE 1 – Le profil de sécurité de signature doit aussi être pris en charge par d'autres entités H.235 (telles que portiers, passerelles et serveurs mandataires H.235).

NOTE 2 – Les bits d'utilisation de clé disponibles dans le certificat pourraient également déterminer le service de sécurité fourni par un terminal (par exemple: non-répudiation affirmée).

Pour l'authentification, il convient que l'utilisateur se serve d'un système de signature à clé publique ou privée. Un tel système offre généralement une intégrité et non-répudiation meilleures de l'appel.

Le présente Recommandation ne définit pas de procédure:

- pour l'enregistrement, la certification et l'attribution de certificat à partir d'un centre de confiance, ainsi que pour l'attribution de clés privées/publiques, pour les services d'annuaire, les paramètres CA spécifiques, l'annulation de certificats, la mise à jour/récupération de paires de clés, de même que d'autres procédures opérationnelles de gestion relatives aux certificats, par exemple la remise de certificats ou de clés publiques/privées et de certificats ainsi que l'installation dans les terminaux.

De telles procédures peuvent être exécutées par des moyens qui ne font partie de la présente annexe.

Les entités de communication concernées ont la capacité de déterminer implicitement l'utilisation du profil de sécurité élémentaire de l'Annexe D ou le présent profil de sécurité à signatures par l'évaluation des identificateurs d'objet de sécurité signalés dans les messages (**tokenOID** et **algorithmOID**; voir également E.18).

E.3 Prescriptions H.323

Les entités H.323 qui mettent en œuvre le présent profil de signature sont supposées prendre en charge les caractéristiques H.323 suivantes:

- la connexion rapide;
- le modèle à routage par portier.

E.4 Services de sécurité

La présente annexe utilise les termes suivants pour décrire les services de sécurité.

- **Authentification seule**: ce service de sécurité du profil de sécurité de signature prend en charge l'authentification de l'utilisateur lorsque celui-ci s'authentifie par une signature numérique correcte d'un élément de données au moyen de la clé privée. On notera que ce service de sécurité n'offre pas de contre-mesures en cas d'opération "couper & coller" arbitraire, de manipulation de message ou d'agression par altération. L'authentification seule peut être utile pour les serveurs mandataires de sécurité qui vérifient l'authenticité du message (authentification de l'origine des données) lors du routage[12] de ce message à une autre destination (le portier, par exemple). Néanmoins, l'authentification seule peut également être appliquée au niveau bond par bond. La procédure III définit ce service de sécurité pour le scénario de bout en bout alors que la procédure II la définit au niveau bond par bond.
- **Authentification et intégrité**: service de sécurité combiné prenant en charge l'intégrité de message en plus de l'authentification de l'utilisateur. L'utilisateur est authentifié par une signature numérique correcte d'un élément de données au moyen de la clé privée. Cela protège en outre contre les altérations. Les deux services de sécurité sont fournis par le même mécanisme de sécurité. L'authentification et l'intégrité combinées ne sont possibles qu'au niveau bond par bond. Ce service de sécurité est défini dans la procédure II.

NOTE – Par l'utilisation des signatures numériques il est possible de prendre en charge un service de sécurité de non-répudiation; cela dépend aussi des réglages des bits d'utilisation de la clé de signature dans le certificat (voir également RFC 2459).

Des techniques asymétriques à signatures numériques peuvent s'appliquer aux niveaux bond par bond et/ou de bout en bout.

Les procédures suivantes sont destinées à être utilisées dans le présent profil:

la procédure II est fondée sur des signatures numériques utilisant une paire de clés privées/publiques pour assurer l'authentification, l'intégrité et la non-répudiation des messages RAS, Q.931 et H.245. Les terminaux peuvent utiliser cette méthode si la non-répudiation et une intégrité élaborée sont requises.

Selon la politique de sécurité, l'authentification peut être unilatérale ou bilatérale, l'authentification/intégrité étant alors appliquée dans les deux sens, ce qui accroît la sécurité. La politique de sécurité relative à un terminal peut permettre l'authentification seule sans calcul de l'intégrité cryptographique (voir E.7).

Les portiers qui détectent une authentification qui n'a pas abouti et/ou une validation de l'intégrité qui n'a également pas abouti dans un message RAS/signalisation d'appel reçu d'un terminal ou d'un portier homologue répondent par un message de refus correspondant indiquant l'absence de sécurité par la mise du motif de refus à **securityDenial**.

Une signalisation H.235 implicite permet d'indiquer l'utilisation de la procédure II et le mécanisme de sécurité appliqué sur la base de la valeur des identificateurs d'objet (voir également E.18) et les champs de message qui ont été remplis. Les identificateurs d'objets sont désignés symboliquement au moyen de lettres (par exemple "A") dans le présent texte.

Le présent profil n'utilise pas les champs ICV H.235; au lieu de cela, les valeurs de contrôle d'intégrité cryptographiques sont placées dans le champs **signature** du jeton **token** du **cryptoSignedToken**.

---

[12] Le routage modifie généralement certaines parties du message; il n'est donc pas possible d'obtenir l'intégrité de bout en bout.

E.5 Signatures numériques avec détails des paires de clés publiques/privées (procédure II)

Il est nécessaire de se conformer aux procédures suivantes si la procédure II est utilisé pour la sécurité au niveau bond par bond:

- Il convient d'utiliser l'algorithme SHA1 ou MD5 avec l'algorithme RSA pour produire la signature numérique. En l'occurrence, la conformité aux systèmes PKCS n° 1 et PKCS n° 7 favorise l'interopérabilité.

  Le champ **CryyptoH323Token** de chaque message RAS/H.225.0 doit contenir les champs suivants:

  – **nestedCryptoToken** contenant un **CryptoToken** contenant le champ **cryptoSignedToken** avec les champs suivants:
    - **tokenOID** mis à:
      – "A" pour indiquer que le calcul d'authentification/intégrité englobe tous les champs du messages RAS/H.225.0 (voir E.9);
      – "B" pour indiquer que le calcul d'authentification/intégrité englobe seulement un sous-ensemble de champs (voir E.8) du message RAS/H.225.0 pour l'authentification seule;
    - **token** contenant les champs:
      – **toBeSigned** contenant le champ **EncodedGeneralToken**, qui est en fait un **ClearToken** avec les champs suivants:
        - **tokenOID** mis à "S" pour indiquer que **ClearToken** est en cours d'utilisation pour l'authentification/intégrité/non-répudiation d'un message.
        - **timeStamp** contenant l'horodateur.
        - **random** contenant un numéro de séquence croissant monotone.
        - **generalID** contenant l'identificateur du destinataire (seulement en cas de monodiffusion).
        - **sendersID** contenant l'identificateur de l'expéditeur.
        - **dhkey**, utilisé pour transmettre les paramètres Diffie-Hellman comme défini dans la présente Recommandation au cours de **Setup** et **Connect.**
          – **halfkey** contenant la clé publique aléatoire d'une des parties;
          – **modsize** contenant DH-prime (voir Tableau D.4);
          – **generator** contenant le DH-group (voir Tableau D.4).

          NOTE 1 – Lorsque le profil de sécurité de signature est utilisé sans le profil de sécurité de cryptage vocal, aucun paramètre Diffie-Hellman ne doit être envoyé; au lieu de cela, les champs **halfkey**, **modsize** et **generator** peuvent être mis à la représentation binaire de 0 pour des raisons de simplicité.

          – **certificate** contenant le certificat numérique de l'expéditeur (voir E.12).
        - **algorithmOID** mis à:
          – "V" pour indiquer l'utilisation de la signature MD5-RSA;
          – "W" pour indiquer l'utilisation de la signature SHA1 RSA.
        - **params** mis à NULL.

- **signature** contenant la signature calculée au moyen de l'algorithme SHA1 ou MD5 RSA sur l'ensemble des champs (si **tokenOID** est "A", voir E.9) ou certains champs déterminants (si **tokenOID** est "B", voir E.8) du message RAS/H.225.0.

Lorsque **tokenOID** est mis à "A" pour la protection des unités H323-UU-PDU canalisées en tunnel, y compris tout le contenu du message H.245, le calcul des signatures doit être fait sur l'ensemble des messages **H.323-UU-PDU** avec l'ensemble des champs conformément à la procédure décrite au E.9. Si **tokenOID** est mis à "B", l'authentification seule de **CryptoToken** est réalisée par l'application de la procédure III (voir E.8).

- L'entité à laquelle la signature est destinée (elle peut être distante d'un ou de plusieurs bonds de niveau application) vérifie cette signature.

NOTE 2 – Le destinataire a la capacité de détecter l'utilisation de la procédure II en évaluant **algorithmOID** dans le jeton de **cryptoSignedToken** (par détection de la présence de "V" ou de "W").

E.6 Procédures de conférence multipoint

Les ponts MCU doivent prendre en charge la répartition sécurisée des certificats sur la demande des terminaux par les commandes H.245 **ConferenceRequest** et **ConferenceResponse** canalisées en tunnel, comme indiqué au 9.1. Cela permet aux terminaux de demander des certificats à d'autres terminaux dans un contexte de conférence multipoint et d'obtenir ainsi avec certitude l'identité des autres participants à la conférence.

ConferenceRequest achemine **requestTerminalCertificate** dont les champs:

- **terminalLabel**: est utilisé comme moyen d'adressage du terminal distant via le pont MCU;
- **certSelectionCriteria**: par lequel l'expéditeur peut demander uniquement des certificats de types donnés;
- **sRandom**: une épreuve aléatoire produite par l'expéditeur demandeur.

ConferenceResponse achemine **terminalCertificateResponse** dont les champs:

- **terminalLabel**: permet d'associer le certificat renvoyé au terminal
- **CertificateResponse**: achemine la réponse du pont MCU avec les champs mis à:
  - **terminalLabel**: identification du terminal distant
  - **certificateResponse**: il s'agit en fait d'une chaîne d'octets ASN.1 codée à partir de **EncodedReturnSig** en tant que:
    - **generalID**: identification du terminal de destination;
    - **responseRandom**: valeur d'épreuve aléatoire produite par le pont MCU;
    - **requestRandom**: **sRandom** reproduit:
    - **certificate**: achemine le certificat renvoyé dans lequel **type** indique le type de certificat en tant qu'identificateur OID et **certificate** achemine le certificat numérique (voir E.12).

E.7 Authentification de bout en bout (procédure III)

La Figure E.1 représente un scénario dans lequel des serveurs mandataires séparent les portiers GK et les points EP et dans lequel deux valeurs CryptoToken sont utilisées pour l'authentification bond par bond ainsi que pour l'authentification de bout en bout et/ou l'intégrité bond par bond. La valeur **CryptoToken** pour l'authentification bond par bond s'applique uniquement au tronçon compris entre deux entités et doit être recalculée pour chaque nouveau tronçon. Par ailleurs, la valeur de **CryptoToken** pour l'authentification de bout en bout est produite une seule fois par le point d'extrémité expéditeur et n'est pas modifiée pendant le transport par des nœuds intermédiaires. Ceux-

ci peuvent valider des signatures et des certificats acheminés dans des **CryptoToken** de bout en bout et devraient acheminer la valeur **CryptoToken** pendant le transport.

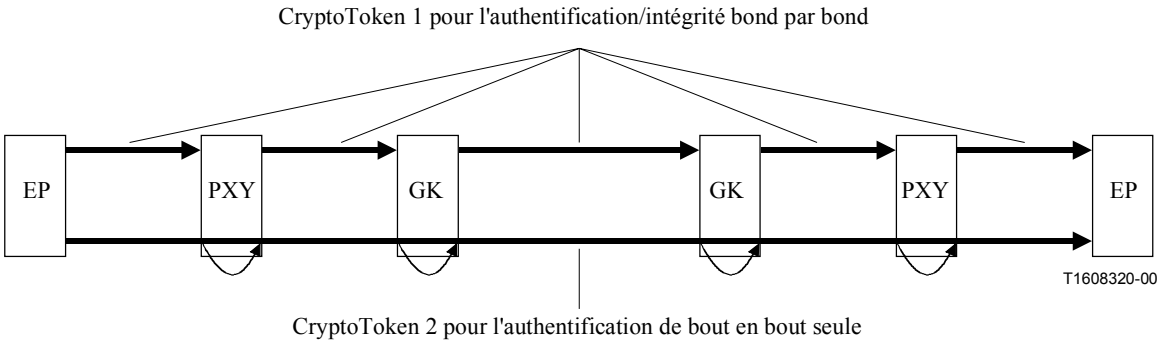


Figure E.1/H.235 – Utilisation simultanée de la sécurité bond par bond et de l'authentification de bout en bout

NOTE 1 – Le serveur mandataire peut être un nœud de réseau séparé comme indiqué dans la Figure E.1 ou peut être situé au même endroit que la fonctionnalité d'une entité H.323, par exemple faire partie du portier.

NOTE 2 – Selon le **tokenOID** signalé, le serveur mandataire a la capacité de déterminer si la valeur **CryptoToken** reçue est destinée au serveur mandataire ("S") ou à un autre destinataire ("R").

NOTE 3 – Vu que les entités intermédiaires changent le contenu du message de signalisation pour chaque tronçon, l'intégrité de bout en bout n'est pas possible.

Pour obtenir une authentification vraie de bout en bout, de part et d'autre des serveurs mandataires H.323 et des éléments de réseau intermédiaires, le point d'extrémité/terminal expéditeur doit calculer une signature numérique de la manière suivante:

Le champ **CryptoH323Token** de chaque message RAS/H.225.0 doit contenir les champs suivants:

- **nestedCryptoToken** contenant un **CryptoToken** qui lui-même contient **cryptoSignedToken** avec les champs suivants:
  - **tokenOID** mis à:
    - "A" pour indiquer que le calcul de l'authentification/intégrité bond par bond englobe tous les champs du message RAS/H.225.0 (voir E.9);
    - "B" pour indiquer que le calcul d'authentification englobe uniquement un sous-ensemble de champs (voir E.8) du message RAS/H.225.0 en vue de l'authentification seule.
- **token** contenant les champs:
  - **toBeSigned** contenant le champ **ClearToken** utilisé avec les champs suivants:
    - **tokenOID** mis à "R" pour indiquer que **ClearToken** est en cours d'utilisation pour l'authentification seule/non-répudiation[13] au niveau de bout en bout;
    - **random** contenant un numéro de séquence croissant monotone;
    - **timeStamp**, facultativement pour une sécurité améliorée seulement lorsque les entités de terminal sont synchronisées;

[13] Le service de sécurité qui sera effectivement appliqué dépend également des bits d'utilisation de la clé dans le certificat.

- **generalID** contenant l'identificateur de point d'extrémité du destinataire (en cas de monodiffusion seulement). Au niveau bond par bond, il s'agit de l'identificateur du bond suivant; au niveau de bout en bout, il s'agit de l'identificateur du point d'extrémité éloigné;
- **sendersID** contient l'expéditeur au point d'extrémité;
- **certificate** contenant le certificat numérique de l'expéditeur, où **type** indique le type de certificat ("V" pour un certificat MD5-RSA ou "W" pour un certificat SHA1-RSA) et **certificate** achemine le certificat proprement dit (voir E.12);
- **dhkey**, utilisé pour acheminer les paramètres Diffie-Hellman spécifiés dans la présente Recommandation au cours de **Setup** et **Connect**:
  - **halfkey** contenant la clé publique aléatoire d'une des parties;
  - **modsize** contenant DH-prime (voir Tableau D.4);
  - **generator** contenant DH-group (voir Tableau D.4).

NOTE 4 – Lorsque le profil de sécurité de signature est utilisé sans le profil de sécurité de cryptage vocal, aucun paramètre Diffie-Hellman ne doit être envoyé; au lieu de cela, **halfkey**, **modsize** et **generator** peuvent être mis à la représentation binaire de 0 pour des raisons de simplicité.

- **token** avec les champs:
  - **algorithmOID** mis à
    - "V" pour indiquer l'utilisation de la signature MD5-RSA;
    - "W" pour indiquer l'utilisation de la signature SHA1-RSA.
  - **params** mis à NULL;
  - **signature** contenant la signature calculée au moyen de l'algorithme SHA1-RSA ou MD5-RSA sur l'ensemble des champs (si **tokenOID** est mis à "A") ou seulement sur certains champs déterminants (si **tokenOID** est mis à "B") du message RAS/H.225.0.

Le serveur mandataire peut vérifier toute signature numérique ou certificat reçu et peut ignorer le message qu'il juge inapproprié compte tenu de la politique locale, ou bien faire suivre le **CryptoToken** qu'il a reçu. Le serveur mandataire doit produire de nouveaux éléments d'information de signalisation H.235 pour la sécurité bond par bond conformément à la procédure II ou III.

Il convient que l'entité qui termine le tronçon – un terminal, par exemple – vérifie les informations de sécurité reçues dans le **CryptoToken** et, selon la présence d'éléments de sécurité de bout en bout, peut évaluer aussi l'information **CryptoToken** de bout en bout. Les procédures de vérification exactes à faire dans un terminal ou une entité H.323 intermédiaire peuvent varier en fonction de la politique locale.

E.8 Authentification seule

Les terminaux peuvent décider de mettre en œuvre l'authentification seule (on utilisant l'identificateur OID "B"). Dans ce cas l'authentificateur est calculé sur un sous-ensemble seulement (**ClearToken** de **CryptoToken**) du message RAS/H.225.0. L'authentification seule peut être utile pour l'authentification vraie de bout en bout (voir E.7). Les champs suivants de la structure **ClearToken** sont utilisés en tant que sous-ensemble:

- **tokenOID**: identificateur d'objet jeton séparé (tokenOID "B") pour l'implémentation d'authentification seule.
- **random**: numéro de séquence croissante monotone.
- **timeStamp**: horodateur.

- **generalID**: identificateur du destinataire (en mode diffusion seulement). Au niveau bond par bond, il s'agit de l'identificateur du bond suivant; au niveau de bout en bout, il s'agit de l'identificateur du point d'extrémité distant.
- **sendersID**: identificateur de l'expéditeur.
- **dhkey**: paramètres Diffie-Hellman. Ces champ et sous-champs sont utilisés uniquement au cours des messages **Setup** et **Connect**.

L'authentificateur est calculé sur **ClearToken** à l'intérieur de **EncodedGeneralToken** (c'est-à-dire **ClearToken**) du champ **token** de **cryptoSignedToken**. La signature numérique sera calculée sur la chaîne binaire codée ASN.1 de **ClearToken**. Avant de calculer la signature numérique, le champ **tokenOID** de **ClearToken** doit être mis à {0 0}.

E.9 Authentification et intégrité

La procédure pour l'authentification et l'intégrité du message sur l'ensemble des champs de message codés ASN.1 (identificateur OID "A") est la suivante:

L'expéditeur du message doit calculer la signature de la manière suivante:

1) mettre la valeur de signature à un modèle par défaut spécifique de longueur fixe (par exemple 1024 bits). Dans cette étape, il faut réserver de l'espace pour la longueur maximale d'une signature numérique, ce qui est possible en raison d'un certificat donné. La configuration binaire exacte importe peu, mais il est préférable de choisir une configuration qui ne survient pas dans la suite du message;
2) coder l'ensemble des messages en ASN.1;
3) localiser la configuration par défaut dans le message codé et l'écraser par des bits zéro[14];
4) calculer la signature numérique à partir du message codé ASN.1 par la méthode indiquée par **algorithmOID** à savoir "V" ou "W" (voir E.10);
5) substituer la configuration par défaut dans le message codé par la valeur correspondant à la signature numérique calculée. Si la signature numérique est plus courte que l'espace réservé, placer des zéros devant les bits de plus fort poids de la valeur de signature.

Le destinataire reçoit les messages et procède ensuite de la manière suivante:

1) décode le message ASN.1;
2) extrait la valeur correspondant à la signature numérique reçue et la conserve dans une variable initiale (SV) locale;
3) recherche et localise la valeur de signature SV dans le message codé reçu;

NOTE – Dans le cas rare où la sous-chaîne de la valeur de signature survient plusieurs fois dans l'ensemble du message, il convient d'itérer les bonds 3 à 6 avec des positions de départ de la recherche différentes.

4) écraser la configuration binaire du message codé avec des zéros;
5) calculer la signature numérique à partir du message codé ASN.1 par la méthode indiquée par **algorithmOID** à savoir "V" ou "W" (voir E.10);
6) comparer la variable SV avec la valeur de signature calculée. Le message est considéré exempt d'erreur et authentique seulement si les valeurs de signature sont identiques; dans ce cas, l'authentification a abouti et la procédure s'arrête;

---

[14] Cela peut sous-entendre quelques essais et quelques erreurs au cas, très rare, où la configuration par défaut survient plusieurs fois dans le message.

7) sinon, répéter les opérations 3) à 7) en rétablissant la variable SV à l'emplacement précédent et en recherchant d'autres concordances. Si aucune des concordances ne donne une comparaison des valeurs de signature correcte, l'authentification n'a pas abouti et le message a été altéré (accidentellement ou intentionnellement) au cours du transport ou pour toute autre raison.

E.10 Calcul de la signature numérique

Au départ du processus de production de la signature numérique, il y une chaîne binaire codée ASN.1 ainsi que le résultat du processus du calcul du résumé du message et la clé privée du signataire. Les détails de la production de la signature numérique dépendent de l'algorithme de signature utilisé; le certificat détermine l'algorithme de signature qu'il convient d'appliquer; lorsque l'extension relative à l'utilisation de la clé figure dans le certificat, le bit **digitalSignature** doit être mis de manière à correspondre à la clé pouvant être utilisée pour la signature. La valeur de signature produite par le signataire est codée sous forme d'une chaîne binaire et acheminée dans le champ **signature**.

Il faudra utiliser la méthode décrite dans [PKCS numéro 1, section E.8.1.1] pour calculer une signature numérique de type RSA au moyen de l'appendice (RSASSA-PKCS1-v1_5-SIGN) et avec les procédures OS2IP, RSASP1, I2OSP et la méthode de codage EMSA-PKCS1-v1_5.

E.11 Vérification de la signature numérique

Le départ du processus de vérification de la signature est le résultat du processus de calcul du résumé du message et la clé publique du signataire. Le destinataire peut obtenir la clé publique correcte pour le signataire par n'importe quel moyen, mais la méthode préférée est celle d'un certificat obtenu dans le champ **certificate** et ensuite validé au moyen du hachage du certificat du signataire. La validation de la clé publique du signataire peut être basée sur le traitement du trajet de certification (RFC 2459). Les détails de la vérification de la signature dépendent de l'algorithme de signature employé.

Il faudra utiliser la méthode dans [PKCS numéro 1, section E.8.1.2] pour vérifier une signature numérique de type RSA au moyen de l'appendice (RSASSA-PKCS1-v1_5-VERIFY) et avec les procédures OS2IP, RSAVP1, I2OSP et la méthode EMSA-PKCS1-v1_5-ENCODE.

E.12 Traitement des certificats

Pour la vérification des signatures numériques, l'entité de réception doit avoir accès au certificat de l'expéditeur, qui est signé par une autorité de certification reconnue (CA, *certification authority*). Il y a différentes possibilités pour le destinataire d'accéder au certificat de l'expéditeur:

- le certificat est inclus dans l'échange de message tel qu'il est décrit dans les procédures II et III;
- le destinataire connaît le certificat; celui-ci a éventuellement été enregistré localement lors d'un échange antérieur;
- plutôt que d'inclure le certificat proprement dit, l'expéditeur fournit une adresse URL où le certificat peut être trouvé. A cet effet, **certificate** contient l'URL et **type** est mis à l'identificateur OID "P";
- le destinataire obtient la certification par un autre moyen qui ne relève pas de la présente Recommandation (par exemple consultation d'annuaire par protocole LDAP).

Les procédures II et III offrent des moyens pour acheminer le certificat numérique. Pour des raisons d'efficacité, les certificats numériques des entités doivent être transmis au maximum une seule fois, à moins qu'ils ne soient déjà disponibles dans les entités (par d'autres moyens qui ne relèvent pas de la présente Recommandation). L'échange de certificats devrait donc survenir uniquement au début de

l'établissement d'une communication: pour le RAS, cela se produit durant la découverte du portier ou, si cette phase est omise, pendant l'enregistrement du portier. Il en est de même en connexion rapide, où le certificat peut être inclus dans les messages de signalisation d'appel initiaux mais peut être omis en toute sécurité dans les messages de signalisation d'appel ultérieurs.

Pour le présent profil de sécurité, il faut utiliser le certificat X.509v3 (1997). D'autres formats de certificat feront l'objet d'un complément d'étude.

E.13 Exemple d'utilisation de la procédure II

Considérons le cas de la Figure E.2 dans lequel chaque entité a sa propre paire de clés publiques ou privées ou son propre certificat. Une entité peut également disposer de plusieurs paires de clés. Dans la figure, un serveur mandataire H.323 sépare le point EP1 du portier GK1.

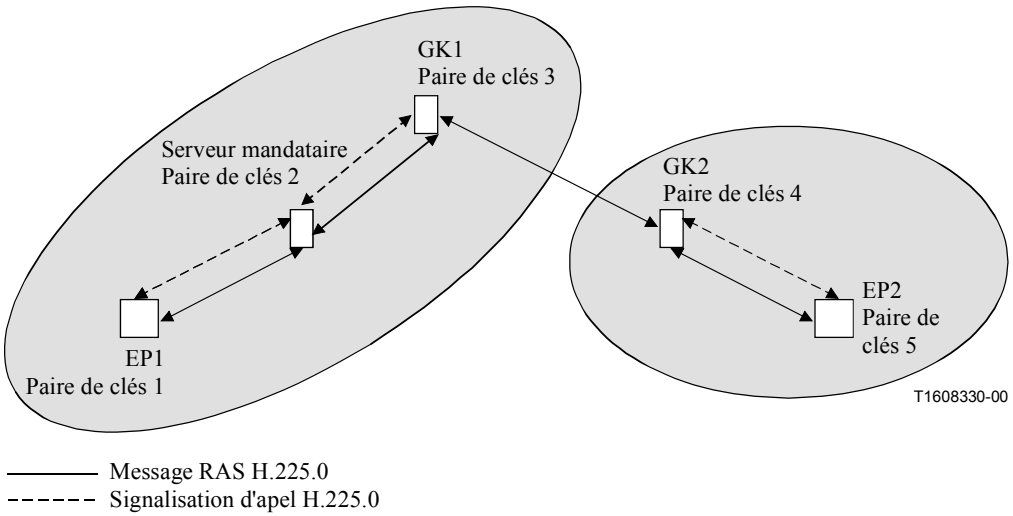


Figure E.2 /H.235 – Exemple de l'utilisation de clés publiques dans un modèle routé de portier à portier

Le serveur mandataire H.323 a un comportement double: d'une part, il met fin à l'authentification et à l'intégrité de chacun de ses tronçons. Il ajoute activement les informations d'authentification/intégrité venant d'être calculées dans les messages RAS sortants d'une manière analogue à celle décrite dans la procédure I de l'Annexe D. Par ailleurs, le serveur mandataire laisse passer les informations de sécurité de bout en bout sans modification. Il peut toutefois vérifier les certificats et/ou signatures numériques reçus en transit.

Ci-dessous, on trouvera l'illustration des détails de procédure pour l'authentification, l'intégrité et la non-répudiation des messages RAS, H.225.0 et H.245.

E.13.1 Authentification, intégrité et non-répudiation des messages RAS

Considérons le cas d'une communication bond par bond dans laquelle le point EP1 souhaite envoyer un message RAS – par exemple un message **ARQ –** au portier GK1. Le point EP1 produit un horodateur et un numéro de séquence qu'il inclut respectivement dans les champs **timeStamp** et **random** avec l'alias du serveur mandataire dans le champ **generalID** et l'identificateur du point EP1 dans **sendersID**. Ces champs sont présents dans le champ **ClearToken** de **EncodedGeneralTokens** présent dans le **token** de **cryptoSignedToken** du champ **CryptoToken** de **cryptoH323Token** du message **ARQ**. Ce **cryptoH323Token** est l'un des (pour le moins) nombreux jetons de la séquence **cryptoTokens**. Le champ **tokenOID** de **cryptoSignedToken** est mis à "A" pour indiquer que tous les champs des messages **ARQ** sont signés. Le **token** de **cryptoSignedToken** a son champ

algorithmOID mis à "V" pour indiquer l'utilisation de la compilation MD5-RSA ou à "W" pour indiquer l'utilisation de l'algorithme SHA1-RSA, et le champ **params** mis à NULL. Le point EP1 calcule ensuite la signature sur la base de l'algorithme de signature en question en utilisant sa propre clé privée. La signature est calculée sur l'ensemble des champs du message **ARQ** lorsque **tokenOID** est mis à "A". Le point EP1 englobe la signature calculée dans **signature** du champ **token** du champ **cryptoSignedToken** de **CryptoToken** qui se trouve dans **cryptoH323Token** du message **ARQ,** et inclut son certificat dans le champ **certificate**.

D'une manière analogue à la communication de bout en bout passant par un serveur mandataire, le point EP1 produit un autre **CryptoToken** contenant une signature numérique qui couvre certains domaines déterminants (voir E.7) dans **ClearToken** du message **ARQ**. Le champ **tokenOID** de **CryptoSignedToken** est mis à "B" pour indiquer l'authentification seule de ce **ClearToken**; il met le champ **tokenOID** de **ClearToken** à "R" pour indiquer l'authentification de bout en bout, remplit les champs **timeStamp**, **random**, **sendersID**, **generalID** (et, s'il s'agit d'un **SETUP/CONNECT,** de **dhkey)**, ainsi que les champs suivants dans **token**: **algorithmOID** à "V" ou "W" pour indiquer l'algorithme de signature, **params** à NULL et **signature** à la signature numérique calculée en fonction des champs **ClearToken**. Le **certificate** achemine le certificat numérique du point EP1. Le message **ARQ** est ensuite envoyé au serveur mandataire.

Lorsqu'il reçoit le message **ARQ**, le serveur mandataire vérifie la signature des jetons qui lui sont adressés (dans ce cas, par exemple, ceux ayant le **tokenOID** "A") sur la base de plusieurs critères, notamment:

- l'actualité de l'horodateur et l'unicité de **random**;
- l'identité de **generalID** et de son propre identificateur;
- les autorisations d'accès pour **sendersID**;
- la concordance de la signature du message **ARQ** et de celle calculée par le portier GK1;
- la vérification des paramètres Diffie-Hellman, en contrôlant si les paramètres "prime" et "generator" à 1024 bits sont corrects. La vérification des paramètres DH est une opération longue qui n'aura lieu que si la politique locale l'exige;
- la vérification du certificat reçu.

Si la vérification de la signature est positive, le serveur mandataire calcule une nouvelle signature qu'il substitue à l'ancienne dans le message **ARQ** avant d'envoyer celui-ci au portier GK1 de la manière suivante: le serveur mandataire remplace les champs **timeStamp**, **random, sendersID** et **generalID** de **ClearToken** (**toBeSigned**) par des valeurs s'appliquant à un tronçon compris entre le serveur mandataire et le portier 1. Le champ **timestamp** contient l'horodateur en vigueur, le champ **random** contient le numéro de séquence croissant monotone suivant pour le tronçon serveur mandataire-portier 1, le champ **sendersID** du serveur mandataire et **generalID** contient un alias du portier GK1. Le serveur mandataire calcule ensuite une nouvelle signature pour ce message **ARQ** en utilisant sa clé privée et l'algorithme de signature, l'introduit dans **signature** de **token** et ajoute son **certificate**. Le serveur mandataire introduit aussi le **CryptoToken** de bout en bout et son **ClearToken** qu'il a reçus dans le nouveau message sortant et passe le message **ARQ** au portier GK1. La signature calculée par le point EP1 sur la base d'une sélection de champs du message **ARQ** (**tokenOID** de "B") et qui n'était pas destinée au serveur mandataire est également transmise sans changement dans le message **ARQ** au portier GK1.

Lorsqu'il reçoit le message **ARQ**, le portier GK1 vérifie les signatures, calcule une nouvelle signature après avoir changé les champs **ClearToken** en **toBeSigned** comme il convient, l'introduit dans le champ **signature**, ajoute son **certificate** et passe le message **Setup** au portier EP2. Ici aussi, le portier GK1 devrait envoyer toute information de bout en bout reçue dans les **CryptoToken** séparés au portier homologue GK2 en plaçant ces informations inchangées dans un **CryptoToken** séparé.

E.13.2 Authentification RAS seule

Considérons le cas d'une communication bond par bond dans laquelle le point EP1 souhaite envoyer un message RAS – par exemple un message **ARQ** – au portier GK1. Le point EP1 produit un horodateur et un numéro de séquence qu'il inclut respectivement dans les champs **timeStamp** et **random** avec l'alias du serveur mandataire dans le champ **generalID** et l'identificateur du point EP1 dans **sendersID**. Ces champs sont présents dans le champ **ClearToken** de **toBeSigned** présent dans le **token** de **cryptoSignedToken** du champ **CryptoToken** de **cryptoH323Token** du message **ARQ**. Le champ **tokenOID** de **cryptoSignedToken** est mis à "B" pour indiquer que seul le sous-ensemble spécifié de champs du message de **ClearToken** est signé. Le **token** de **cryptoSignedToken** a son champ **algorithmOID** mis à "V" pour indiquer l'utilisation de la compilation MD5-RSA ou à "W" pour indiquer l'utilisation de l'algorithme SHA1-RSA, et le champ **params** mis à NULL. Le point EP1 calcule ensuite la signature sur la base de l'algorithme de signature en question en utilisant sa clé privée. La signature est calculée sur les champs **ClearToken** spécifiés dans **ARQ**. Le point EP1 englobe la signature calculée dans **signature** du champ **token** du champ **cryptoSignedToken** de **CryptoToken** qui se trouve dans **cryptoH323Token** du message **ARQ,** et inclut son **certificate**.

D'une manière analogue, le point EP1 produit une autre signature numérique pour l'authentification de bout en bout qui couvre certains champs **ClearToken** dans un **CryptoToken** individuel du message **ARQ**. Cette signature numérique (identifiée par le **tokenOID** "V" ou "W") est incluse. Le message **ARQ** est ensuite envoyé au serveur mandataire.

Lorsqu'il reçoit le message **ARQ**, le serveur mandataire vérifie la signature des jetons qui lui sont adressés (dans ce cas, par exemple, ceux ayant le **tokenOID** "B") sur la base de plusieurs critères, notamment:

- l'actualité de l'horodateur et l'unicité de **random**;
- l'identité de **generalID** et de son propre identificateur;
- les autorisations d'accès pour **sendersID**;
- la concordance de la signature du message **ARQ** et de celle calculée par le portier GK1;
- la vérification du certificat reçu.

Si la vérification de la signature est positive, le serveur mandataire calcule une nouvelle signature qu'il substitue à l'ancienne dans le message **ARQ** avant d'envoyer celui-ci au portier GK1 de la manière suivante: le serveur mandataire remplace les champs **timeStamp**, **random, sendersID** et **generalID** de **ClearToken** (**toBeSigned**) par des valeurs s'appliquant au tronçon compris entre le serveur mandataire et le portier 1. Le champ **timestamp** contient l'horodateur en vigueur, le champ **random** contient le numéro de séquence croissant monotone suivant pour le tronçon serveur mandataire-portier 1, et **generalID** contient l'alias du portier GK1. Le serveur mandataire calcule ensuite une nouvelle signature pour ce **ClearToken** en utilisant sa clé privée et l'algorithme de signature MD5-RSA ou SHA1-RSA (**algorithmOID** à "V" ou à "W"), l'introduit dans **signature** de **token** de **cryptoSignedToken**, ajoute son **certificate** et passe le message **ARQ** au portier GK1. La signature calculée par le point EP1 sur la base d'une sélection de champs **ClearToken** du message **ARQ** (**tokenOID** de "B") et qui n'était pas destinée au serveur mandataire est également transmise sans changement dans le message **ARQ** au portier GK1.

Lorsqu'il reçoit le message **ARQ**, le portier GK1 vérifie les signatures, calcule une nouvelle signature après avoir changé les champs **ClearToken** en **toBeSigned** comme il convient, l'introduit dans le champ **signature** et passe le message **Setup** au portier EP2. Les informations de signature de bout en bout du point EP1 sont jointes sans changement dans le message **Setup**.

E.13.3 Authentification, intégrité et non-repudiation de message H.225.0

La procédure s'appliquant aux messages H.225.0 est la même que celle utilisée pour les messages RAS. La seule différence réside dans la nécessité d'identifier pour chaque message H.225.0 l'ensemble de champs qu'il convient de signer lorsque **tokenOID** est mis à "B".

E.13.4 Authentification et intégrité de message H.245

Considérons le cas où le point EP1 souhaite envoyer un message H.245 – un message **TerminalCapabilitySet** par exemple – au point EP2. Le point EP1 détermine si un message H.225.0 est en attente d'envoi au serveur mandataire. Si c'est le cas, le message H.245 est canalisé en tunnel dans ce message H.225.0. Les champs de message H.225.0 sont mis aux valeurs indiquées précédemment pour la transmission des messages H.225.0. Etant donné que le message H.245 est canalisé en tunnel, les champs de **h323-uu-pdu** du message **h323-UserInformation** sont mis comme suit:

- **h323-message-body** au type de message H.225.0 message type en cours de transmission.
- **h245Tunneling** à TRUE.
- **h245Control** contenant la chaîne d'octets de pont PDU H.245.

Toutefois, si aucun message H.225.0 est en attente d'envoi, le message H.245 est canalisé en tunnel dans un message H.225.0 **facility** ad hoc. Les champs **h323-uu-pdu** du message **h323-UserInformation** sont mis comme suit:

- **h323-message-body** à **facility** qui contient:
  - **reason** à **undefinedReason**;
  - **tokens** et **cryptoTokens** comme pour tout message H.225.0.
- **h245Tunneling** à TRUE.
- **h245Control** contenant la chaîne d'octets pont PDU H.245.

Le message **facility** est ensuite transmis par le point EP1 au serveur mandataire.

Dans les deux cas (message H.225.0 en attente d'envoi ou utilisation d'un message H.225.0 **facility** ad hoc), le serveur mandataire vérifie la signature destinée à cet effet (dans ce cas, représenté par **tokenOID** "A") lorsqu'il reçoit le message. Ensuite, si une transmission de message H.225.0 transmission est en attente pour le tronçon serveur mandataire-portier GK1, le message H.245 est canalisé en tunnel dans ce message; sinon, il est canalisé en tunnel dans un message H.225.0 **facility** ad hoc. Comme c'est le cas pour la transmission de tout message H.225.0, une nouvelle signature est calculée pour celui-ci avant qu'il ne soit transmis du serveur mandataire au portier GK1. La signature qui avait été envoyée du point EP1 au serveur mandataire et qui n'était pas destiné au serveur mandataire est transmis sans modification par le serveur mandataire au portier GK1.

Le présent paragraphe propose un résumé de la manière et des moyens par lesquels le profil de sécurité effectue la sécurisation des divers messages de signalisation H.323.

E.14 Compatibilité avec le contexte H.235 Version 1

Bien que ces profils de sécurité soient mis au point dans le contexte H.235 version 2 [H.235 (2000)], il est possible de les appliquer dans un contexte H.235 version 1 [H.235 (1998)] moyennant quelques modifications mineures. Un destinataire a la capacité de détecter la présence de la version du protocole H.235 de l'expéditeur par l'évaluation des identificateurs d'objet du profil de sécurité (voir E.18).

Implémentations H.235 version 1 [H.235 (1998)]:

- ne pas attribuer de valeur à ou ne pas évaluer **sendersID** de **ClearToken**.

E.15 Comportement en multidiffusion

Les messages H.225.0 multidiffusés tels que **GRQ** et **LRQ** doivent comporter un **CryptoToken** conformément aux procédures II et III lorsque aucune valeur n'est attribuée à **generalID**. Lorsque de tels messages sont monodiffusés, le message doit comporter un **CryptoToken**.

E.16 Liste des messages de signalisation sécurisés

E.16.1 Message RAS H.225.0

Message RAS H.225.0	Champs de signalisation H.235	Authentification seule	Authentification et intégrité	Non-répudiation
tous	cryptoTokens	procédure II/III	procédure II/III	procédure II/III

NOTE – Dans le cas des messages monodiffusés, la procédure II ou III doit être appliquée avec utilisation des champs de sécurité de **CryptoToken**.

E.16.2 Signalisation d'appel H.225.0

Message de signalisation d'appel H.225.0	Champs de signalisation H.235	Authentification seule	Authentification et intégrité	Non-répudiation
Alerting-UUIE, CallProceeding-UUIE, Connect-UUIE, Setup-UUIE, Facility-UUIE, Progress-UUIE, Information-UUIE, ReleaseComplete-UUIE	cryptoTokens	Procedure II/III	Procedure II/III	Procedure II/III

E.17 Utilisation des identificateurs sendersID et generalID

Le paramètre **ClearToken** contient les champs d'identificateurs **sendersID** et **generalID**. Lorsque l'information d'identification est disponible, la valeur de l'identificateur **sendersID** est celle de l'identificateur du portier (GKID) pour le message provenant du portier, et celle de l'identificateur du point d'extrémité (EPID) pour les messages provenant du point d'extrémité. Lorsque l'information d'identification est disponible, la valeur de l'identificateur **generalID** est celle de l'identification du portier (GKID) pour les messages provenant du point d'extrémité, et celle de l'identificateur du point d'extrémité, (EPID) pour les messages provenant du portier. Lorsque l'information d'identification n'est pas disponible ou lorsque la diffusion/multidiffusion est ambiguë, le champ est absent ou contient une chaîne de zéros. Le Tableau E.2 ci-après résume la situation:

Tableau E.2/H.235 – Identificateurs d'objet utilisés dans l'Annexe E

Message	sendersID	GeneralID
Unicast **GRQ**	**EPID** si disponible, autrement **NULL**	**GKID**
Multicast **GRQ**	**EPID** si disponible, autrement **NULL**	
GCF, GRJ	**GKID**	**EPID** si disponible, autrement **NULL**
Initial **RRQ**		**GKID**
RCF	**GKID**	**EPID**
RRJ	**GKID**	

Tableau E.2/H.235 – Identificateurs d'objet utilisés dans l'Annexe E *(fin)*

Message	sendersID	GeneralID
URQ, UCF, URJ, BRQ, BCF, BRJ, DRQ, DCF, DRJ, NSM, RIP, SCI, SCR, XRS (EP-to-GK)	**EPID**	**GKID**
URQ, UCF, URJ, BRQ, BCF, BRJ, DRQ, DCF, DRJ, NSM, RIP, SCI, SCR, XRS (GK-to-EP)	**GKID**	**EPID**
ARQ, IRQ, RAI	**EPID**	**GKID**
ACF, ARJ, BCF, LCF, LRJ, IRR, IRQ, RAC, LCF, LRJ, IACK, INAK	**GKID**	**EPID**
Unicast **LRQ** (EP-to-GK)	**EPID**	**GKID**
Unicast **LRQ** (GK-to-GK)	**GKID**	**GKID**
Multicast **LRQ**	**EPID**	
NOTE – GKID désigne l'identificateur du portier, EPID désigne l'identificateur du point d'extrémité. L'espace vide indique une chaîne d'identification manquante ou nulle.		

E.18 Liste des identificateurs d'objet

Le Tableau E.3 contient tous les identificateurs OID qui ont été mentionnés (voir aussi [OIW] et [WEBOIDs]. Il y a des identificateurs d'objet pour H.235v1 [H.235 (1998)] et pour H.235v2 (2000)].

Tableau E.3/H.235 – Identificateurs d'objet utilisés dans l'Annexe E

Désignation d'identificateur d'objet	Valeur(s) d'identificateur d'objet	Description
"A"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 1} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 1}	Utilisé dans la procédure II pour le CryptoToken-tokenOID indiquant que la signature englobe **tous** les champs du message RAS/H.225.0 (authentification et intégrité).
"B"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 2} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 2}	Utilisé dans la procédure II pour le CryptoToken-tokenOID indiquant que la signature englobe un **sous-ensemble** des champs du message RAS/H.225.0 (ClearToken) pour terminaux à authentification seulement (sans intégrité).
"P"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 4} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 4}	Utilisé dans la procédure II ou III pour indiquer que le **certificate** achemine un URL.
"R"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 3} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 3}	Utilisé dans la procédure II pour le ClearToken-tokenOID indiquant que le ClearToken est en cours d'utilisation pour l'authentification/intégrité de bout en bout.
"S"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 7} {itu-t (0) recommendation (0) h (8) 235 version (0) 1 7}	Utilisé dans la procédure II, ce tokenOID indique l'authentification, l'intégrité et la non-répudiation du message.

Tableau E.3/H.235 – Identificateurs d'objet utilisés dans l'Annexe E *(fin)*

Désignation d'identificateur d'objet	Valeur(s) d'identificateur d'objet	Description
"V"	{iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 4}	Utilisé dans la procédure II comme OID d'algorithme indiquant l'utilisation de la signature numérique MD5 RSA.
"W"	{iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5}	Utilisé dans la procédure II comme OID d'algorithme indiquant l'utilisation de la signature numérique SHA1 RSA.

APPENDICE I

Détails d'implémentation H.323

I.1 Méthodes de bourrage cryptographique

Il existe une description d'emprunt cryptographique dans [**Schneier**], p. 191 et 196. Les Figures I.1 à I.5 illustrent cette technique.

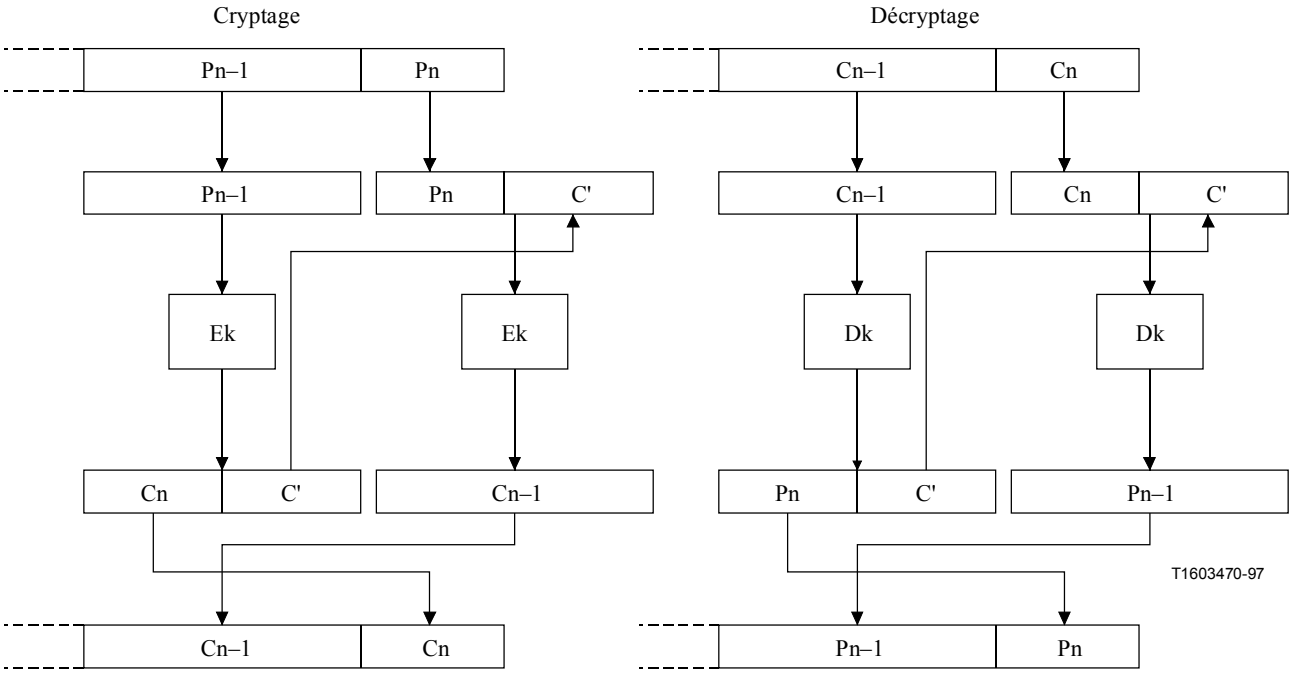


Figure I.1/H.235 – Interception cryptographique en mode ECB

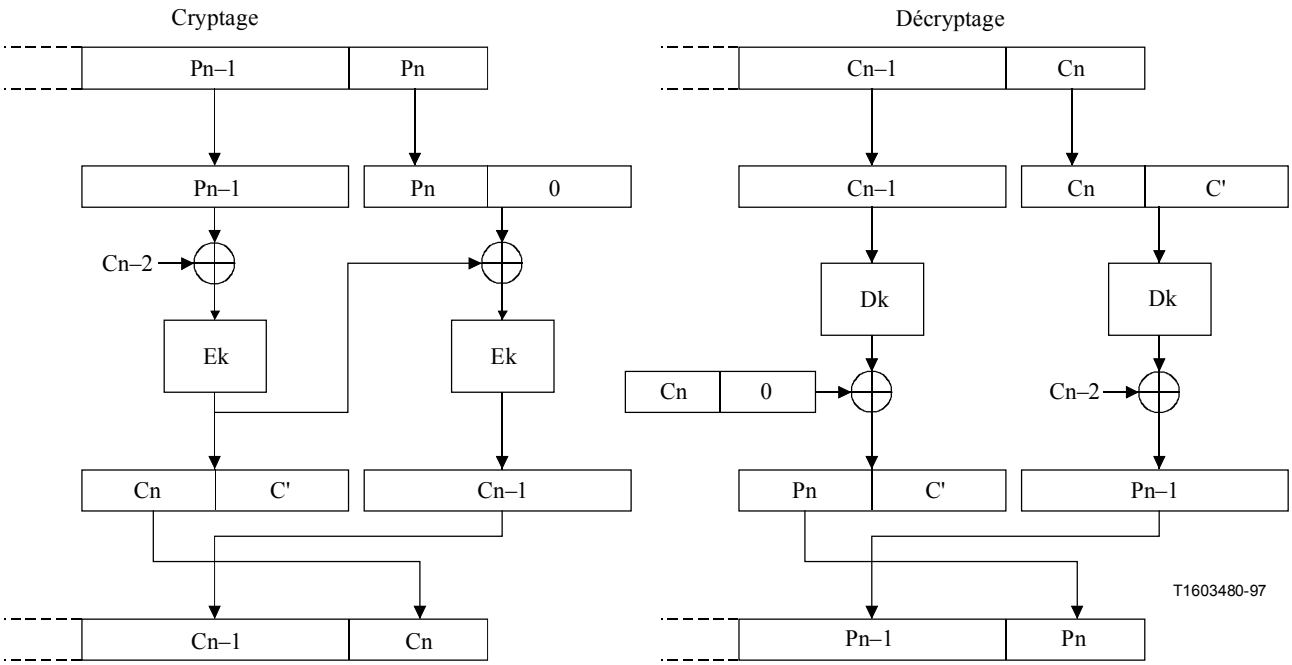


Figure I.2/H.235 – Interception cryptographique en mode CBC

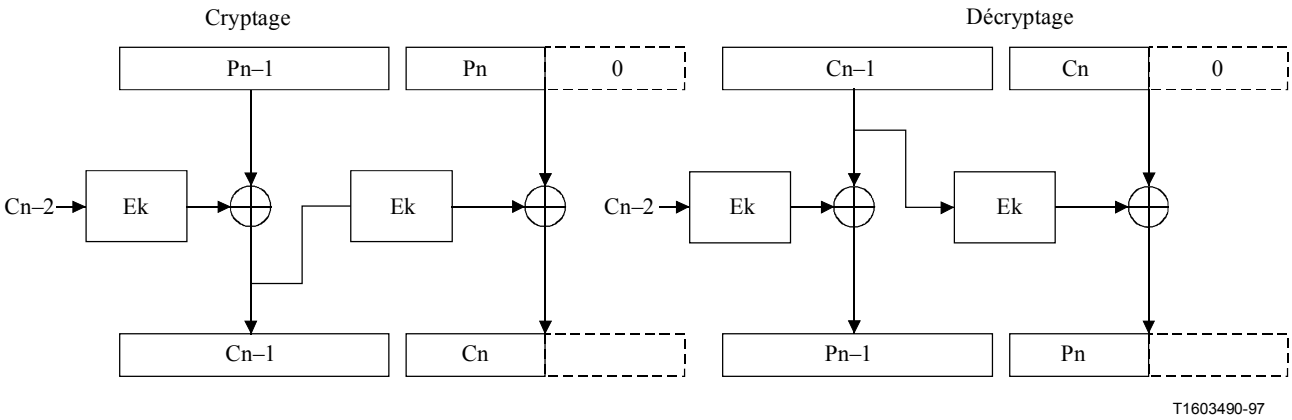
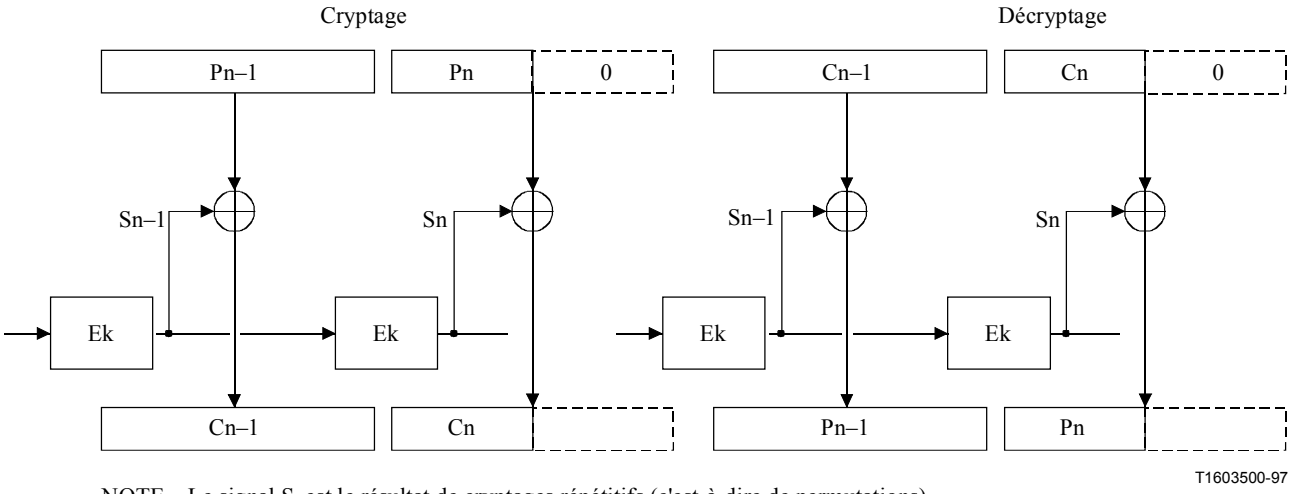


Figure I.3/H.235 – Bourrage de zéros en mode CFB



NOTE – Le signal S_i est le résultat de cryptages répétitifs (c'est-à-dire de permutations) du vecteur d'initialisation.

Figure I.4/H.235 – Bourrage de zéros en mode OFB

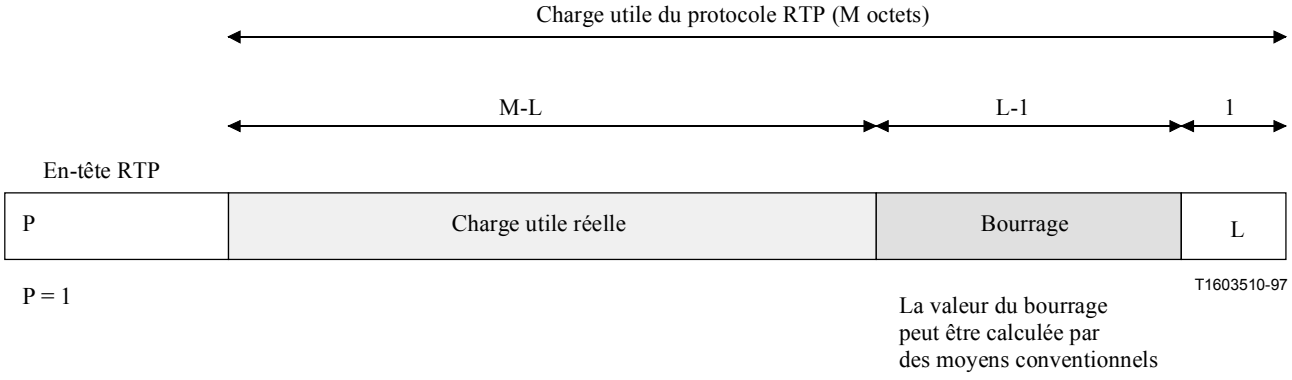


Figure I.5/H.235 – Bourrage tel que prescrit par le protocole RTP

I.2 Nouvelles clés

Les procédures décrites au 8.5/H.323 sont appliquées par un pont de conférence afin d'éjecter un participant d'une conférence. Le point maître peut produire de nouvelles clés de chiffrage pour les canaux logiques (et ne pas les distribuer au correspondant éjecté); cette méthode peut être utilisée afin d'empêcher le correspondant éjecté de surveiller les flux médias.

I.3 Eléments crédibilisés H.323

En général, les ponts de conférence, les passerelles et les portiers (s'ils implémentent le modèle à routage par portier) sont des éléments crédibilisés pour ce qui est du secret des communications par le canal de commande. Si le canal d'établissement des connexions (H.225.0) est sécurisé *et* acheminé par l'entremise du portier, l'on doit également s'y fier. Si l'un de ces éléments H.323 doit fonctionner avec les flux médias (c'est-à-dire pour un mixage, un transcodage), ils doivent alors, par définition, être aussi considérés comme crédibles pour le secret des communications médias.

Les serveurs tampons ou pare-feu (bien que ne constituant pas des éléments spécifiques de l'UIT-T H.323) peuvent aussi être crédibilisés, car ils terminent des connexions et peuvent tout à fait avoir à manipuler les messages et les flux médias.

I.4 Exemples d'implémentation

Les sous-paragraphes suivants décrivent des exemples d'implémentation qui pourraient être développés dans le cadre de H.235. Ils ne sont pas destinés à prendre le pas sur les nombreuses autres possibilités proposées dans la présente Recommandation. Ces paragraphes visent plutôt à donner des exemples plus concrets d'utilisation dans le cadre de l'UIT-T H.323.

I.4.1 Jetons

Le présent paragraphe décrira un exemple d'utilisation de jetons de sécurité afin d'occulter ou de masquer les informations d'adressage de destination. Le scénario donné en exemple est un point d'extrémité qui souhaite établir une communication avec un autre point d'extrémité utilisant son alias notoire. Plus précisément, le réseau H.323 se compose d'un point d'extrémité A, d'un portier, d'une passerelle avec le POTS et d'un poste téléphonique B, comme illustré dans la Figure I.6.

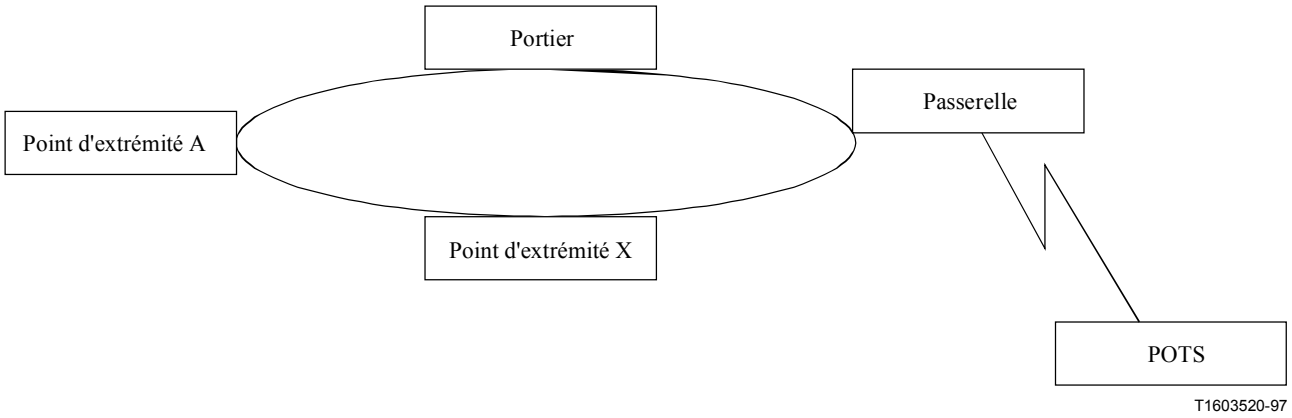


Figure I.6/H.235 – Jetons

Actuellement, un réseau H.323 peut fonctionner de façon analogue à un réseau téléphonique avec identification de l'appelant. Ce scénario illustre une situation dans laquelle le *appelé* ne souhaite pas divulguer son adresse physique tout en acceptant l'établissement de l'appel. Cela peut être important dans les passerelles POTS-H.323, lorsque le numéro téléphonique de destination peut devoir rester privé.

Supposons que le point A essaye d'appeler le point POTS-B et que celui-ci ne souhaite pas divulguer au point A son numéro de téléphone selon le plan E.164. (La façon dont cette politique est établie est hors du domaine d'application de cet exemple.)

- le point EPA enverra une demande ARQ à son portier pour résoudre l'adresse du poste POTS, tel que représenté par son alias/passerelle. Le portier reconnaîtra cette adresse comme un alias "privé", sachant que pour réaliser la connexion il doit renvoyer l'adresse de la passerelle avec le POTS. (Ce cas est analogue à celui du renvoi d'adresse d'une passerelle H.320 si un point d'extrémité H.320 est appelé par un point H.323.);
- dans le message ACF renvoyé, le portier renvoie l'adresse de la passerelle avec le POTS, comme prévu. Les informations d'adressage qui sont nécessaires pour appeler le poste distant (c'est-à-dire le numéro de téléphone) sont renvoyées dans un jeton codé dans le message ACF. Ce jeton crypté contient le numéro E.164 réel (de téléphone) du poste, qui ne peut pas être déchiffré ni compris par l'appelant (c'est-à-dire le point EPA);
- le point d'extrémité envoie à la passerelle tête de ligne (dont l'adresse de signalisation d'appel a été renvoyée par le message ACF) un message SETUP contenant le ou les jetons opaques qu'il a reçus dans l'ACF;

- dès qu'elle reçoit le message SETUP, la passerelle envoie sa demande ARQ à son portier, y compris tous jetons reçus dans le message SETUP;
- le portier est en mesure de déchiffrer le ou les jetons et de renvoyer le numéro de téléphone dans le message ACF.

Une partie de la notation ASN.1 d'une structure de jeton est montrée ci-dessous à titre d'exemple, avec description du contenu des champs. L'on suppose que l'on utilise le paramètre **cryptoEncodedGeneralToken** pour y insérer le numéro de téléphone crypté.

Une mise en œuvre peut choisir un identificateur d'objet jeton, **tokenOID**, pour indiquer que ce jeton contient le numéro de téléphone E.164. La méthode particulière qui sera utilisée pour coder ce numéro de téléphone (par exemple une norme DES à 56 bits) sera incluse dans la définition "ENCRYPT' contenue dans l'identificateur d'algorithme, **algorithmOID**.

```
CryptoToken::= CHOICE
{
        cryptoEncodedGeneralToken SEQUENCE   -- Jeton à usage général ou à application spécifique
        {
                tokenOID  OBJECT IDENTIFIER,
                ENCRYPTED { EncodedGeneralToken }
        },
.
.
. [abbreviated text]
.

}
```

Le message **CryptoToken** sera transmis dans le message SETUP (du point EPA à la passerelle) et les messages **ARQ** (de la passerelle au portier) seront transmis comme indiqué ci-dessus. Après avoir déchiffré le jeton (numéro de téléphone), le portier en transmet la version en clair dans le paramètre **clearToken**.

I.4.2 Utilisation des jetons dans les systèmes H.323

L'utilisation des jetons **CryptoH323Tokens** tels qu'ils sont acheminés dans les messages RAS a donné lieu à une certaine confusion. Il y a deux grandes catégories de jetons **CryptoH323Tokens**: celle des jetons utilisés pour les procédures H.235 et celle des jetons utilisés d'une manière spécifique à l'application. Il convient d'utiliser ces jetons conformément aux règles suivantes:

- tous les jetons définis dans H.235 (par exemple **cryptoEPPwdHash**, **cryptoGKPwdHash**, **cryptoEPPwdEncr**, **cryptoGKPwdEncr**, **cryptoGKCert**, et **cryptoFastStart**) doivent être utilisés conformément aux procédures et avec les algorithmes définis dans la présente Recommandation);
- les jetons spécifiques aux applications et les jetons propriétaires doivent utiliser pour leurs échanges, le jeton **nestedcryptoToken**;
- tout jeton **nestedcryptoToken** doit utiliser un jeton **tokenOID** (identificateur d'objet) qui l'identifie sans équivoque.

I.4.3 Utilisation de la valeur aléatoire H.235 dans les systèmes H.323

La valeur aléatoire qui est transmise dans une séquence xRQ/xCF entre les points d'extrémité et les portiers peut être actualisée par le portier. Comme indiqué au B.4.2, cette valeur aléatoire peut être rafraîchie dans tout message xCF en vue de l'utilisation dans des messages xRQ subséquents partant du point d'extrémité. Etant donné qu'il y a possibilité de perte des messages RAS (y compris xCF/xRJ), la valeur aléatoire actualisée peut également se perdre. La reprise à partir d'une telle situation peut être la réinitialisation du contexte de sécurité, mais le soin en est laissé à la réalisation.

Les implémentations qui nécessitent l'utilisation de plusieurs demandes RAS en suspens seront limitées par l'actualisation des valeurs aléatoires utilisées dans toute authentification. Si l'actualisation de cette valeur se produit à chaque réponse à une demande, les demandes parallèles sont impossibles. Une solution éventuelle consiste à disposer d'une "fenêtre" logique au cours de laquelle une valeur aléatoire reste constante. Il s'agit d'une question à résoudre au plan de l'implémentation.

I.4.4 Mot de passe

Dans cet exemple, l'on suppose que l'utilisateur est abonné au service de portier (c'est-à-dire qu'il se trouve dans la zone de celui-ci) et qu'il possède un identificateur d'abonnement et un mot de passe correspondants. Cet utilisateur va s'enregistrer auprès du portier en utilisant son identificateur d'abonnement (tel qu'il a été transmis dans un identificateur d'alias H.323) et en chiffrant une chaîne d'épreuve rédhibitoire qui lui sera présentée par le portier. Ce processus suppose que le portier connaît également le mot de passe associé à l'identificateur d'abonnement. Le portier authentifiera l'utilisateur en vérifiant que la chaîne d'épreuve a été correctement chiffrée.

La procédure d'enregistrement avec authentification par portier sera la suivante pour cet exemple:

1) si le point d'extrémité utilise une demande **GRQ** pour découvrir un portier, un des alias contenus dans le message se trouvera dans l'identificateur d'abonnement (sous forme d'identificateur **H323ID**). Le message **authenticationcapability** contiendra un mécanisme d'authentification (**AuthenticationMechanism**) par codage de mot de passe (**pwdSymEnc)** et les identificateurs d'algorithme (**algorithmOID**) seront paramétrés de façon à indiquer l'ensemble complet des algorithmes de cryptage pris en charge par le point d'extrémité. (Par exemple, l'un de ces algorithmes sera la norme DES à 56 bits en mode EBC);

2) le portier répondra à ce message par une confirmation **GCF** (en supposant qu'il reconnaît l'alias) acheminant un élément **tokens** contenant un seul jeton en clair, **ClearToken**. Celui-ci se composera de deux parties: une épreuve rédhibitoire (**challenge**) et un pointeur temporel **timeStamp**. L'épreuve **challenge** sera codée sur 16 octets (pour prévenir les agressions par répétition, le jeton en clair **ClearToken** contiendra un élément **timeStamp**). Le mode d'authentification **authenticationmode** sera mis à **pwdSymEnc** et l'identificateur d'algorithme **algorithmOID** indiquera l'algorithme de cryptage requis par le portier (par exemple norme DES à 56 bits en mode EBC).

   Si le portier ne prend en charge aucun des identificateurs d'algorithme **algorithmOID** indiqués dans la demande **GRQ**, il répondra par un rejet **GRJ** contenant une cause **GatekeeperRejectReason** égale à **ressourceUnavailable**;

3) l'application du point d'extrémité tentera alors de s'enregistrer auprès du (d'un des) portier(s) ayant répondu par une confirmation **GCF**, en envoyant une demande **RRQ** contenant un élément **cryptoEPPwdEncr** dans le paramètre **cryptoTokens**. Cet élément **cryptoEPPwdEncr** contiendra l'identificateur de l'algorithme de cryptage **algorithmOID** convenu lors de l'échange de messages **GRQ/GCF**, ainsi que l'épreuve rédhibitoire cryptée.

   La clé de cryptage est construite sur la base du mot de passe de l'utilisateur, au moyen de la procédure décrite au 10.3.2. La "chaîne" d'octets résultante sera alors utilisée comme clé DES pour chiffrer l'épreuve rédhibitoire **challenge**;

4) lorsque le portier reçoit l'épreuve rédhibitoire dans la demande **RRQ**, il la compare à une épreuve rédhibitoire déjà chiffrée à l'identique, afin d'authentifier l'utilisateur requérant. Si les deux chaînes chiffrées ne correspondent pas, le portier répond par un message de rejet **RRJ** avec la cause **RegistrationRejectReason** mise à la valeur **securityDenial**. Si les chaînes correspondent, le portier envoie une confirmation **RCF** au point d'extrémité;

5) si le portier reçoit une demande **RRQ** qui ne contient pas d'élément **cryptoTokens** acceptable, il doit répondre par un rejet **RRJ** avec la cause **GatekeeperRejectReason** mise à la valeur **discoveryRequired**. Le point d'extrémité, dès qu'il reçoit ce message **RRJ**, peut

exécuter la recherche qui permettra au couple portier/point d'extrémité d'échanger une nouvelle épreuve rédhibitoire. On notera que le message **GRQ** peut être envoyé en mode point à point au portier.

I.4.5 Sécurité IPSEC

En général, la méthode IPSEC [**13/IPSEC**] peut être utilisée pour assurer l'authentification et, facultativement, la confidentialité (c'est-à-dire le cryptage) dans la couche IP de façon transparente à tout protocole (applicatif) exploité dans les couches supérieures. Le protocole applicatif n'a pas besoin d'être mis à jour pour permettre cette opération; seule la politique de sécurité à chaque extrémité doit correspondre.

Par exemple, pour tirer le meilleur parti de la sécurité IPSEC pour une simple communication point à point, le scénario ci-après peut être suivi:

1) le point d'extrémité appelant et son portier détermineront par le protocole RAS la politique prescrivant l'utilisation de la sécurité IPSEC (authentification et, facultativement, confidentialité). Avant l'envoi du premier message RAS du point d'extrémité au portier, le démon ISAKMP/Oakley situé au point d'extrémité négociera les services de sécurité à utiliser pour les paquets à destination et en provenance de l'accès notoire du canal RAS. Une fois la négociation achevée, le canal RAS fonctionne exactement comme s'il n'avait pas été sécurisé. Au moyen de ce canal sécurisé, le portier informera le point d'extrémité de l'adresse et du numéro d'accès du canal de signalisation d'appel se trouvant au point d'extrémité appelé;
2) après avoir obtenu l'adresse et le numéro d'accès du canal de signalisation d'appel, le point d'extrémité appelant met à jour dynamiquement sa politique de sécurité afin de demander la sécurité IPSEC souhaitée à cette adresse pour cette paire protocole/accès. Ensuite, lorsque le point d'extrémité appelant tentera de se mettre en contact avec cette paire adresse/accès, les paquets seront mis en file d'attente pendant l'exécution d'une négociation par routine ISAKMP/Oakley entre les points d'extrémité. A l'achèvement de cette négociation, une association de sécurité IPSEC existera pour cette paire adresse/accès et la signalisation Q.931 pourra commencer;
3) lors de l'échange des messages Q.931 SETUP et CONNECT, les points d'extrémité peuvent négocier l'utilisation de la sécurité IPSEC pour le canal H.245. Cela permettra aux points d'extrémité de remettre à jour dynamiquement leurs bases de données pour politique de sécurité IPSEC et d'imposer l'utilisation de cette politique sur cette connexion;
4) comme dans le canal de signalisation d'appel, une négociation ISAKMP/Oakley transparente se déroulera avant qu'un quelconque paquet H.245 soit émis. L'authentification effectuée par cet échange ISAKMP/Oakley sera la tentative initiale d'une authentification d'utilisateur à utilisateur. Elle établira un canal (probablement) sécurisé entre les deux utilisateurs, permettant de négocier les caractéristiques du canal audio. Si, à la suite d'un dialogue interpersonnel, l'un des utilisateurs n'est pas satisfait de l'authentification, différents certificats peuvent être choisis et l'échange ISAKMP/Oakley peut être répété;
5) après chaque authentification ISAKMP/Oakley H.245, de nouvelles données de clé sont échangées pour le canal audio en protocole RTP. Ces données sont distribuées par le maître sur le canal H.245 sécurisé. Comme le protocole H.245 est défini de façon que le maître distribue les données de clés multimédias sur le canal H.245 (afin de permettre des communications multipoints), il n'est pas recommandé d'utiliser la méthode IPSEC pour le canal RTP.

Un canal H.245 crypté peut poser un problème pour les serveurs tampons ou les pare-feu NAT car les numéros d'accès attribués dynamiquement sont acheminés dans le protocole H.245. Pour fonctionner correctement, de tels pare-feu devront déchiffrer, modifier et rechiffrer le protocole. C'est pourquoi le canal logique "de sécurité" a été introduit dans l'UIT-T H.245. Si ce canal est

utilisé, le canal H.245 peut rester non sécurisé; l'authentification et la production de clés seront effectuées avec le canal logique "de sécurité". La signalisation par canal logique permettra de protéger ce canal par la méthode IPSEC et la clé à secret utilisée dans le canal logique "de sécurité" servira à protéger la synchronisation (**EncryptionSync**) distribuée par le maître sur le canal H.245.

I.4.6 Prise en charge des services de réalisation spécialisés

Les serveurs spécialisés représentent une fonction supplémentaire importante dans l'ensemble de l'environnement multimédia de type H.323. Le serveur BES fournit, par exemple, des services pour l'authentification de l'utilisateur, pour l'autorisation de service ainsi que pour la comptabilité, la taxation et la facturation et d'autres services. Dans un modèle simple, le portier peut fournir de tels services, mais dans une architecture décomposée, il ne peut pas toujours le faire, soit parce qu'il n'a pas nécessairement accès aux bases de données BES, soit parce qu'il fait partie d'un domaine administratif différent. Par ailleurs, le terminal et l'utilisateur ne connaissent généralement pas leur serveur BES.

La Figure I.7 représente un scénario comportant un terminal multimédia (par exemple un dispositif SASET), un portier et un serveur BES. La manière exacte dont le serveur BES communique avec le portier ne relève pas de H.323. Plusieurs méthodes et protocoles peuvent être utilisés: la technologie RADIUS (voir RFC 2138), considérée comme l'une des plus importantes, est couramment utilisée par de nombreux fournisseurs de services.

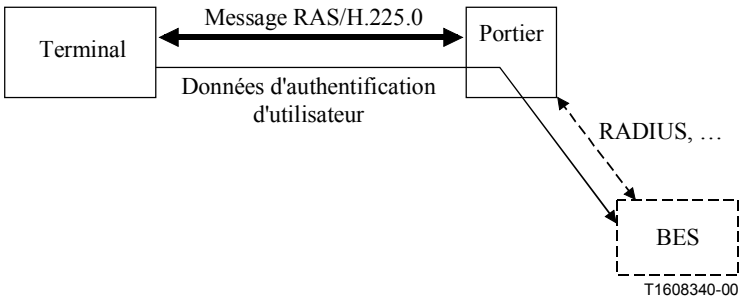


Figure I.7/H.235 – Scénario avec serveur spécialisé

Un portier qui prend en charge des services BES devrait proposer au moins les deux modes suivants:

1) Le mode **default mode**, dans lequel le terminal ne connaît pas le serveur BES où il faut une relation de confiance avec le portier. Le terminal envoie les données d'authentification de l'utilisateur sous forme cryptée (**cryptoEncryptedToken**) au portier; celui-ci les décrypte, en extrait les informations d'authentification d'utilisateur et les envoie au serveur BES. Le cryptage à mot de passe de **ClearToken** est effectué en appliquant à **CryptoToken** un secret distinct qui est connu du terminal et du portier. La clé de cryptage pourrait être obtenue à partir du mot de passe au moyen duquel le terminal s'enregistre de manière sécurisée auprès du portier.

   CryptoToken achemine **cryptoEncryptedToken** dans lequel **tokenOID** est mis à "M" pour indiquer le mode BES par défaut; **token** contient:

   - **algorithmOID** indiquant l'algorithme de cryptage; "Y" (DES56-CBC), "Z" (3DES-ocbc); voir D.11.
   - **params** est inutilisé.
   - **encryptedData** est mis à la représentation en octets du **ClearToken** crypté.

Le **ClearToken** contient en tant que **password** les données d'authentification de l'utilisateur. Les informations **ClearToken** protégées pourraient être le mot de passe ou un code PIN, une identification de l'utilisateur, un numéro de carte à prépaiement ou un numéro de carte de crédit. Le **timestamp** est mis à l'heure du terminal, **random** contient un numéro de séquence croissant monotone, **sendersID** est mis à l'identificateur du terminal et **generalID** à l'identificateur du portier. La valeur initiale de l'algorithme de cryptage doit être maintenue constante; elle ne peut pas faire partie du secret attribué au moment de l'abonnement.

NOTE – Le **ClearToken** n'est pas transmis.

2) Le mode **RADIUS mode**, dans lequel le serveur BES et l'utilisateur du terminal ont un secret commun et dans lequel le portier ne doit pas être "de confiance" pour l'authentification du mode en question. Le portier achemine simplement une épreuve RADIUS reçue du serveur BES dans une épreuve *Access-Challenge* au terminal et envoie la réponse de l'utilisateur sous forme de réponse RADIUS dans une demande *Access-Request* en sens inverse. Le terminal et le portier négocient cette capacité d'épreuve/réponse de **radius** dans **AuthenticationBES** de **AuthenticationMechanism** pendant la découverte du portier.

   Lorsqu'il reçoit un message RADIUS *Access-Challenge* contenant une épreuve, le portier introduit l'épreuve à 16 octets dans le champs **challenge** de **ClearToken** lorsqu'il interroge le terminal avec un message **GCF** ou tout autre message RAS. Le **tokenOID** "K" de **ClearToken** indique une épreuve RADIUS.

   Le terminal peut ensuite présenter l'épreuve à l'utilisateur et attendre la réponse entrée. Le terminal doit répondre au moyen d'un messager RAS dans lequel la réponse figure dans le champ **challenge** de **ClearToken**. Le **tokenOID** "L" de **ClearToken** indique une réponse à RADIUS.

Le Tableau I.1 contient tous les identificateurs OID mentionnés.

Tableau I.1/H.235 – Identificateurs d'objet utilisés au I.4.6

Désignation d'identificateur d'objet	Valeur d'identificateur d'objet	Description
"K"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 1}	indique une épreuve RADIUS dans clearToken
"L"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 2}	indique une réponse RADIUS (acheminée dans le champ challenge) de ClearToken
"M"	{itu-t (0) recommendation (0) h (8) 235 version (0) 2 3}	indique le mode BES par défaut avec mot de passe protégé dans ClearToken

APPENDICE II

Détails d'implémentation H.324

A étudier.

APPENDICE III

Autres détails d'implémentation pour la série H

A étudier.

APPENDICE IV

Bibliographie

[Daemon]

– DAEMON (J.): Cipher and Hash function design, Ph.D. Thesis, Katholieke Universiteit Leuven, Mars 1995.

[IPSEC]

– MAUGHAN (D.), SCHERTLER (M.), SCHNEIDER (M.), TURNER (J.): Internet Security Association and Key Management Protocol (ISAKMP), draft-ietf-ipsec-isakmp-08.text, *Internet Engineering Task Force*, 1997.

[ISO | IEC 14888-3]

– *Technologies de l'information – Techniques de sécurité – Signatures digitales avec appendice – Partie 3: Mécanismes fondés sur certificat*.

[PKCS]

– PKCS #1 v2.0: RSA Cryptography Standard; RSA Laboratories; 1er octobre 1998; http://www.rsa.com/rsalabs/pubs/PKCS/index.html.

– PKCS #7: Cryptographic Message Syntax Standard, An RSA Laboratories Technical Note, Version 1.5, révisée 1er novembre 1993; http://www.rsa.com/rsalabs/pubs/PKCS/index.html

[RTP]

– SCHULZRINNE (H.), CASNER (S.), FREDERICK (R.), JACOBSON (V.): RTP: A transport Protocol for Real-Time Applications, RFC 1889, *Internet Engineering Task Force*, 1996.

[Schneier]

– SCHNEIER (B.): Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition, John Wiley & Sons, Inc., 1995.

[TLS]

– DIEKS (T.), ALLEN (C.): The TLS Protocol Version 1.0, RFC 2246, *Internet Engineering Task Force*, 1999.

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	**Systèmes audiovisuels et multimédias**
Série I	Réseau numérique à intégration de services
Série J	Transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	RGT et maintenance des réseaux: systèmes de transmission, de télégraphie, de télécopie, circuits téléphoniques et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	Réseaux de données et communication entre systèmes ouverts
Série Y	Infrastructure mondiale de l'information et protocole Internet
Série Z	Langages et aspects informatiques généraux des systèmes de télécommunication

Genève, 2001