Union internationale des télécommunications

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

J.1012

(04/2020)

SÉRIE J: RÉSEAUX CÂBLÉS ET TRANSMISSION DES SIGNAUX RADIOPHONIQUES, TÉLÉVISUELS ET AUTRES SIGNAUX MULTIMÉDIAS

Accès conditionnel et protection – Solutions d'accès conditionnel et de gestion des droits numériques intégrées interchangeables

Interface commune intégrée pour les solutions CA/DRM interchangeables; conteneur CA/DRM, chargeur, interfaces et révocation

Recommandation UIT-T J.1012

Recommandation UIT-T J.1012

Interface commune intégrée pour les solutions CA/DRM interchangeables; conteneur CA/DRM, chargeur, interfaces et révocation

Résumé

La Recommandation UIT-T J.1012, qui fait partie d'une publication en plusieurs parties sur l'interface commune intégrée pour les solutions de type accès conditionnel/gestion des droits numériques (CA/DRM) interchangeables, porte sur le conteneur, le chargeur, les interfaces et la révocation pour les solutions CA/DRM.

Cette Recommandation UIT-T, qui est une transposition de la norme ETSI GS ECI 001-3, est le fruit d'une collaboration entre la CE 9 de l'UIT-T et l'ETSI ISG ECI. Des modifications ont été apportées aux § 2, 7.7.2.5.2, 9.4.4.6.2, 9.4.6.1, 9.5.2.2, 9.8.1, 9.8.2, 10.2 et I.2 ainsi qu'à la bibliographie et il a été nécessaire d'apporter quelques corrections rédactionnelles supplémentaires.

Historique

Edition	Recommandation	Approbation	Commission d'études	ID unique*
1.0	UIT-T J.1012	23-04-2020	9	11.1002/1000/13573

Mots clés

CA, DRM, échange.

* Pour accéder à la Recommandation, reporter cet URL http://handle.itu.int/ dans votre navigateur Web, suivi de l'identifiant unique, par exemple http://handle.itu.int/11.1002/1000/11830-en.

AVANT-PROPOS

L'Union internationale des télécommunications (UIT) est une institution spécialisée des Nations Unies dans le domaine des télécommunications et des technologies de l'information et de la communication (ICT). Le Secteur de la normalisation des télécommunications (UIT-T) 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.

Le respect de cette Recommandation se fait à titre volontaire. Cependant, il se peut que la Recommandation contienne certaines dispositions obligatoires (pour assurer, par exemple, l'interopérabilité et l'applicabilité) et considère que la Recommandation est respectée lorsque toutes ces dispositions sont observées. Le futur d'obligation et les autres moyens d'expression de l'obligation comme le verbe "devoir" ainsi que leurs formes négatives servent à énoncer des prescriptions. L'utilisation de ces formes ne signifie pas qu'il est obligatoire de respecter la Recommandation.

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 avait 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 développeurs de consulter la base de données des brevets du TSB sous http://www.itu.int/ITU-T/ipr/.

TABLE DES MATIÈRES

Introduction

La présente Recommandation UIT-T[1], qui est une transposition de la norme [b-ETSI GS ECI 001-3] de l'ETSI, est le fruit d'une collaboration entre la CE 9 de l'UIT-T et l'ETSI ISG ECI. Des modifications ont été apportées aux § 2, 7.7.2.5.2, 9.4.4.6.2, 9.4.6.1, 9.5.2.2, 9.8.1, 9.8.2, 10.2 et I.2 ainsi qu'à la bibliographie et il a été nécessaire d'apporter quelques corrections rédactionnelles supplémentaires.

L'objectif de la présente Recommandation est de faciliter l'interopérabilité et la concurrence en matière de services de communications électroniques et, en particulier, sur le marché de la radiodiffusion et des appareils audiovisuels. Toutefois, d'autres technologies sont disponibles et peuvent également être appropriées et efficaces en fonction de la situation dans les États Membres.

La protection des services et des contenus grâce à l'accès conditionnel (CA) et à la gestion des droits numériques (DRM) joue un rôle essentiel dans le domaine en plein essor de la radiodiffusion et de la diffusion large bande numériques. Elle comprend la distribution de contenus haute définition (HD) et ultra haute définition (UHD) à destination de différents types d'équipements des locaux d'abonnés (**CPE**)[2] afin de protéger les modèles commerciaux des propriétaires de contenus et des fournisseurs de services, notamment les radiodiffuseurs et les opérateurs de télévision à péage. Alors que les systèmes CA sont principalement axés sur la protection des contenus distribués par des réseaux unidirectionnels, utilisés habituellement dans le contexte de la radiodiffusion, les systèmes DRM proviennent d'environnements de réseaux bidirectionnels et autorisent l'accès d'utilisateurs authentifiés aux contenus sur des dispositifs certifiés, avec en général des expressions de droits sur des contenus riches. En pratique, il n'est pas possible d'établir systématiquement une distinction claire entre les fonctionnalités CA et DRM. De ce fait, la présente Recommandation utilise l'expression systèmes CA/DRM.

Les solutions CA/DRM mises en œuvre actuellement, qu'elles soient intégrées ou prennent la forme de matériels séparables, entraînent souvent des restrictions d'usage pour les fournisseurs de services/plates-formes d'une part et les consommateurs d'autre part. Ces derniers se trouvent ainsi dépendants du réseau concerné, des fournisseurs de services et de contenus et du type d'**Équipement CPE** (adapté à la radiodiffusion numérique classique, à la télévision utilisant le protocole Internet (TVIP) ou aux services over-the-top (OTT)). Alors que les **Équipements CPE** dotés de fonctionnalités CA ou DRM intégrées propres à une plate-forme lient les clients à un opérateur de plate-forme donné, les modules matériels séparables permettent d'utiliser un **Équipement CPE** de particulier, par exemple des boîtiers-décodeurs et des téléviseurs numériques intégrés (iDTV). En raison de leur facteur de forme et de leur coût, les modules matériels séparables ne sont pas adaptés à l'évolution de la demande, notamment à la consommation de contenus protégés sur tablettes et dispositifs mobiles et aux déploiements sensibles aux coûts.

De ce fait, les technologies existantes restreignent la liberté de nombreux acteurs du marché des contenus numériques multimédias. Les avancées technologiques permettent désormais de mettre au point des solutions CA/DRM logicielles innovantes. Parce qu'elles allient une interopérabilité maximale et un niveau de sécurité élevé, ces solutions devraient répondre aux futures demandes du marché, permettre l'arrivée de nouveaux acteurs et offrir un choix plus large aux consommateurs en matière de consommation de contenus via des connexions de radiodiffusion et large bande.

Les consommateurs ont intérêt à pouvoir continuer à utiliser, après un déménagement ou un changement de fournisseur de réseau, les **Équipements CPE** qu'ils ont achetés pour leur propre usage et à s'en servir afin de bénéficier des services de différents portails vidéo commerciaux. La mise en oeuvre, à l'intérieur des **Équipements CPE**, de mécanismes CA et DRM interchangeables basés sur

---

[1] Plusieurs domaines nécessitant des développements supplémentaires ont été identifiés dans l'Appendice II.

[2] Dans le texte de la présente Recommandation, on utilise des caractères gras pour les termes dont la définition est propre au contexte de l'interface commune intégrée et peut différer de l'usage courant.

une architecture de sécurité appropriée est en mesure de répondre à ces attentes. Ce n'est qu'en proposant des solutions axées sur l'interchangeabilité conviviale et flexible des systèmes CA et DRM – associée à un environnement de sécurité perfectionné – qu'il sera possible d'éviter une fragmentation accrue du marché des **Équipements CPE** et d'encourager la concurrence.

L'opérateur de plate-forme a intérêt à ce que la technologie de sécurité puisse être déployée de manière flexible et gérée facilement sur différents réseaux et tous les types de dispositifs. L'intégration transparente de systèmes de sécurité de pointe au sein des dispositifs existants offre des opportunités commerciales uniques.

L'**Écosystème ECI** spécifié dans la présente Recommandation, au titre de la publication en plusieurs parties relative aux interfaces **ECI**, présente des attributs importants, tels que la flexibilité et l'évolutivité associées à sa mise en œuvre logicielle, l'interchangeabilité étant propice à une solution évolutive et à l'innovation. Il est également applicable à des contenus distribués par différents types de réseaux, notamment la radiodiffusion numérique classique, la TVIP et les services OTT. La spécification d'un écosystème **ECI** ouvert, propice au développement du marché, constitue la base de l'interchangeabilité des systèmes CA et DRM dans les **Équipements CPE**, au coût le plus bas possible pour les consommateurs, et présente l'avantage de n'imposer aux fabricants de systèmes CA ou DRM que très peu de restrictions en matière de conception de produits cibles destinés au marché de la télévision à péage.

En complément de la Partie 4 de cette publication en plusieurs parties, qui porte sur la machine virtuelle, et de la Partie 5, qui porte sur la sécurité évoluée, la présente Recommandation, qui constitue la Partie 3, spécifie l'ensemble des éléments nécessaires qui sont essentiels au téléchargement et à l'interchangeabilité des clients CA/DRM (**Clients ECI**) et à leur environnement d'exécution (**Hôte ECI**) dans un environnement de confiance, y compris la communication avec les entités fonctionnelles nécessaires via des API, qui sont spécifiées en détail.

Recommandation UIT-T J.1012

Interface commune intégrée pour les solutions CA/DRM interchangeables; conteneur CA/DRM, chargeur, interfaces et révocation

1 Domaine d'application

L'architecture du **Système ECI** est définie dans la Recommandation [UIT-T J.1011]. Voir également [b-ETSI GS ECI 001-1]. Le système **ECI** repose sur les prescriptions énoncées dans la Recommandation [UIT-T J.1010]. Voir également [b-ETSI GS ECI 001-2]. La présente Recommandation spécifie les fonctionnalités de base d'un **Écosystème ECI** et traite en détail du conteneur CA/DRM, du chargeur, des interfaces et de la révocation. Voir également [b-Illgner]. L'innovation et l'avantage majeurs de l'**Écosystème ECI** par rapport aux systèmes déployés actuellement résident dans son architecture de chargement et d'interchangeabilité des systèmes CA/DRM entièrement logicielle qui évite la présence de modules matériels séparables. Les conteneurs logiciels assurent un environnement sécurisé ("carré") pour les noyaux CA ou DRM, appelés ci-après **Clients ECI**, et leurs instances individuelles de **Machine virtuelle**. Les interfaces de programmation d'application (API) nécessaires et pertinentes entre les **Clients ECI** et l'**Hôte ECI** garantissent le fonctionnement de multiples **Clients ECI** dans un environnement d'exploitation sécurisé et totalement isolé du reste du micrologiciel de l'**Équipement CPE**. Elles sont spécifiées en détail. L'installation et l'interchangeabilité d'un **Hôte ECI** et de multiples **Clients ECI** incombent au chargeur **ECI** chargé initialement avec un chargeur de puce. L'**Hôte ECI** et les **Clients ECI** sont téléchargés via le carrousel de données fondé sur un réseau de radiodiffusion vidéo numérique (DVB) pour les services de radiodiffusion et/ou via des mécanismes basés sur IP sur un serveur en cas d'accès large bande. Ce processus, intégré dans un environnement sécurisé et de confiance, met en place une hiérarchie de confiance pour l'installation et l'échange de l'**Hôte ECI** et des **Clients ECI**, qui dresse une protection efficace contre les attaques visant l'intégrité des données et contre les attaques par substitution. C'est pourquoi l'**Écosystème ECI** intègre un mécanisme de sécurité évoluée qui s'appuie sur le traitement efficient et avancé de mots de contrôle (CW) spécifiés en tant que **Blocs d'échelle de clés** intégrés dans un matériel utilisant un système sur puce afin d'assurer la sécurité maximale nécessaire à la conformité **ECI**. Les fonctions de sécurité évoluée propres aux interfaces **ECI** jouent également un rôle essentiel dans le processus de rechiffrement des contenus protégés stockés et/ou d'exportation de contenus protégés vers un dispositif externe conforme **ECI** ou non. Un système micro DRM évolué fournit les fonctionnalités nécessaires et fait partie intégrante de ce concept. Les fonctionnalités de sécurité évoluée jouent également un rôle en cas de révocation d'un **Équipement CPE** ou d'un **Client ECI** particulier. Les API concernées sont spécifiées dans la présente Recommandation, mais la sécurité évoluée est traitée en détail dans les Recommandations [UIT-T J.1014] et [UIT-T J.1015]. Voir également [b-ETSI GS ECI 001-5-1] et [b-ETSI GS ECI 001-5-2].

Plusieurs API caractérisent l'**Écosystème ECI** et garantissent la communication avec les entités pertinentes associées telles que les chargeurs **ECI**, l'importation et exportation de contenus protégés, la sécurité évoluée, le chiffrement et le déchiffrement, les moyens de stockage locaux et l'insertion de filigranes. Il existe d'autres API pour l'interface homme-machine des **Clients ECI** ou un lecteur de **Carte à puce** en option.

Le changement de **Clients ECI** est effectué soit à l'initiative de l'**Utilisateur**, soit à la demande d'un **Opérateur** s'il convient de procéder à des mises à jour. Deux **Clients ECI** au minimum sont pris en charge ainsi que deux autres supplémentaires si le stockage local sur un enregistreur vidéo personnel est possible ou en cas d'exportation.

Les paragraphes suivants de la présente Recommandation fournissent des spécifications détaillées:

Le système de certificats **ECI** est traité dans le § 5 consacré aux différents types de **Certificats** (**Certificats de chargeur d'hôte ECI**, de **chargeur de client ECI** et d'**opérateur ECI**), qui aborde

la définition de ces **Certificats** et la **Liste de révocation** qui leur est associée, leur organisation en chaînes et la structure du **Certificat racine**.

Le **Chargeur d'hôte ECI** est traité au § 6, qui traite du processus de chargement de l'**Hôte ECI**, décrit le stockage des images, la vérification de leur authenticité par l'**Équipement CPE** (à l'aide des données d'authentification fournies par l'**Autorité de confiance ECI**) et leur activation subséquente. Y sont spécifiés le format de fichier ainsi que le protocole de transport et la révocation propre à l'**Opérateur** des **Images de l'hôte ECI**.

Le paragraphe 7 couvre les spécifications détaillées relatives au **Chargeur de client ECI** fondées sur la capacité de l'**Hôte ECI** à télécharger, stocker et activer des **Images de clients ECI** et les données associées. Le processus de chargement des **Clients ECI** peut être décomposé en plusieurs étapes allant de leur découverte jusqu'à leur téléchargement et leur initialisation, ce qui permet de procéder au téléchargement avec des données provenant du flux de radiodiffusion ou d'Internet.

Le paragraphe 8 contient les spécifications détaillées de la révocation, notamment les fonctionnalités permettant l'exclusion sélective de la fourniture de services à destination des **Équipements CPE** sur la base du statut attribué par l'**Autorité de confiance ECI** au matériel **CPE**, à l'**Hôte ECI**, à d'autres **Opérations de plate-forme** et à des **Clients ECI** chargés.

Les spécifications détaillées des interfaces des **Clients ECI** figurent au § 9, qui présente de façon exhaustive les spécifications nécessaires à l'écosystème **ECI**, les API des ressources générales de l'**Hôte ECI**, des ressources de l'**Hôte ECI** propres aux interfaces **ECI**, des ressources de déchiffrement de l'**Hôte ECI**, des ressources de rechiffrement de l'**Hôte ECI**, des ressources liées à la protection des contenus et des ressources en rapport avec le transfert entre **Clients ECI**.

Enfin, le paragraphe 10 s'attache aux fonctionnalités obligatoires et facultatives de l'**Hôte ECI**.

La présente spécification **ECI** principale s'applique uniquement à la réception et au traitement subséquent des contenus contrôlés par un système d'accès conditionnel et/ou de gestion des droits numériques et chiffrés par le fournisseur de services.

La présente Recommandation ne couvre pas les contenus non contrôlés par un système à accès conditionnel et/ou DRM.

Elle doit être utilisée en association avec un cadre contractuel (accord de licence), des règles de conformité et de fiabilité, ainsi que des accords relatifs au processus de certification approprié sous le contrôle d'une autorité de confiance, non assujettis à des spécifications techniques telles que celles stipulées par les spécifications de groupe **ECI**. Certains de ces aspects fondamentaux figurent dans une annexe informative de [b-ETSI GS ECI 001-6], concernant l'environnement sécurisé, qui précise les corrélations et les mécanismes techniques relatifs à un environnement de confiance.

2 Références

Les Recommandations UIT-T et autres références suivantes contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions de la présente Recommandation. Au moment de la publication, les éditions indiquées étaient en vigueur. Les Recommandations et autres références étant sujettes à révision, les utilisateurs de la présente Recommandation sont invités à rechercher la possibilité d'appliquer les éditions les plus récentes des Recommandations et autres références énumérées ci-dessous. Une liste des Recommandations UIT-T en vigueur est publiée périodiquement. La référence à un document figurant dans la présente Recommandation ne donne pas à ce document, en tant que tel, le statut d'une Recommandation.

[UIT-T J.1010] Recommandation UIT-T J.1010 (2016), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Cas d'utilisation et exigences.*

[UIT-T J.1011] Recommandation UIT-T J.1011 (2016), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Architecture, définitions et vue d'ensemble.*

[UIT-T J.1013] Recommandation UIT-T J.1013 (2020), *Interface commune intégrée pour les solutions CA/DRM interchangeables; machine virtuelle.*

[UIT-T J.1014] Recommandation UIT-T J.1014 (2020), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Sécurité évoluée – Fonctionnalités propres aux interfaces ECI.*

[UIT-T J.1015] Recommandation UIT-T J.1015 (2020), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Sécurité évoluée – bloc d'échelle de clés.*

[UIT-T T.871] Recommandation UIT-T T.871 (2011), *Technologies de l'information – Compression numérique et codage des images fixes à modèle continu: format d'échange de fichier JPEG (JFIF).*

[ISO/CEI 23001-7] ISO/CEI 23001-7:2015, *Technologies de l'information – Technologies des systèmes MPEG – Partie 7: Cryptage commun des fichiers au format de fichier média de la base ISO.*

[ISO/CEI 23009-1] ISO/CEI 23009-1:2014, *Technologies de l'information -- Diffusion en flux adaptatif dynamique sur HTTP (DASH) – Partie 1: Description de la présentation et formats de segment des médias.*

[ISO/CEI 13818-1] ISO/CEI 13818-1:2007, *Technologies de l'information – Codage générique des images animées et du son associé – Partie 1: Systèmes.*

[NIST Block 2001] National Institute of Standards and Technology, 2001, *Recommendation for Block Cipher Modes of Operation Methods and Techniques*. <https://www.nist.gov/publications/recommendation-block-cipher-modes-operation-methods-and-techniques>

[NIST FIPS 197] NIST U.S. FIPS PUB 197 (FIPS 197) (2001), *Advanced Encryption Standard (AES).*

[ISO/CEI 21320] ISO/CEI 21320, *Technologies de l'information – Fichier conteneur de document – Partie 1: Données de base.*

[IETF RFC 4122] IETF RFC 4122 (juillet 2015), *A Universally Unique IDentifier (UUID) URN Namespace.*

[CEN EN 50221] CEN EN 50221 (1997), *Common Interface Specification for Conditional Access and other Digital Video Broadcasting Decoder Applications.*

[ETSI TS 102 006] ETSI TS 102 006, *Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems.*

[ETSI EN 301 192] ETSI EN 301 192, *Digital Video Broadcasting (DVB); DVB specification for data broadcasting.*

[ETSI TR 101 202] ETSI TR 101 202, *Digital Video Broadcasting (DVB); Implementation guidelines for Data Broadcasting.*

[ISO/CEI 13818-6] ISO/CEI 13818-6, *Technologies de l'information – Codage générique des images animées et du son associé – Partie 6: Extensions pour DSM-CC.*

[ETSI EN 300 468] ETSI EN 300 468, *Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems.*

[ETSI TS 101 162] ETSI TS 101 162, *Digital Video Broadcasting (DVB); Allocation of identifiers and codes for Digital Video Broadcasting (DVB) systems.*

[ETSI TS 101 211] ETSI TS 101 211, *Digital Video Broadcasting (DVB); Guidelines on implementation and usage of Service Information (SI).*

[IETF RFC 768] IETF RFC 768, *User Datagram Protocol (UDP).*

[IETF RFC 791] IETF RFC 791, *Internet Protocol (IP).*

[IETF RFC 793] IETF RFC 793, *Transmission Control Protocol (TCP).*

[IETF RFC 1034] IETF RFC 1034, *Domain names – Concepts and Facilities.*

[IETF RFC 1035] IETF RFC 1035, *Domain names – Implementation and Specification.*

[IETF RFC 8200] IETF RFC 8200, *Internet Protocol, Version 6 (IPv6) Specification.*

[IETF RFC 1123] IETF RFC 1123, *Requirements for Internet Hosts – Application and Support.*

[IETF RFC 952] IETF RFC 952, *DOD Internet Host Table Specification,*

[ISO/CEI 7816-1] ISO/CEI 7816-1, *Cartes d'identification – Cartes à circuit(s) intégré(s) à contacts – Partie 1: Caractéristiques physiques. Informations générales.*

[ISO/CEI 7816-2] ISO/CEI 7816-2, *Cartes d'identification – Cartes à circuit intégré – Partie 2: Cartes à contacts – Dimensions et emplacements des contacts.*

[ISO/CEI 7816-3] ISO/CEI 7816-3, *Cartes d'identification – Cartes à circuit intégré – Partie 3: Cartes à contacts – Interface électrique et protocoles de transmission.*

[ETSI TS 103 205] ETSI TS 103 205 (V1.2.1) (2015), *Digital Video Broadcasting (DVB); Extensions to the CI Plus™ Specification*.

[ISO/CEI 7816-5] ISO/CEI 7816-5, *Cartes d'identification – Cartes à circuit intégré – Partie 3: Enregistrement des fournisseurs d'application.*

[ISO/CEI 7810] ISO/CEI 7810, *Cartes d'identification – Caractéristiques physiques.*

[ISO/CEI 23001-9] ISO/CEI 23001-9:2014, *Technologies de l'information – Technologies des systèmes MPEG – Partie 9: Cryptage commun des flux de transport de contenu MPEG-2.*

[ETSI TS 103 285] ETSI TS 103 285:2015, *Digital Video Broadcasting (DVB); MPEG-DASH Profile for Transport of ISO BMFF Based DVB Services over IP Based Networks.*

[ISO/CEI 14496-12] ISO/CEI 14496-12:2015, *Technologies de l'information – Codage des objets audiovisuels – Partie 12: Format ISO de base pour les fichiers médias.*

[ETSI ETR 289] ETSI ETR 289 (1996), *Digital Video Broadcasting (DVB); Support for use of scrambling and Conditional Access (CA) within digital broadcasting systems*.

[ETSI TS 103 127] ETSI TS 103 127, *Digital Video Broadcasting (DVB); Content Scrambling Algorithms for DVB-IPTV Services using MPEG2 Transport Streams*.

[ETSI TS 100 289] ETSI TS 100 289, *Digital Video Broadcasting (DVB); Support for use of the DVB Scrambling Algorithm version 3 within digital broadcasting systems.*

[IETF RFC 7230] IETF RFC 7230 (2014), *Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing.*

[IETF RFC 7231] IETF RFC 7231 (2014), *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content.*

[IETF RFC 5246] IETF RFC 5246 (2008), *The Transport Layer Security (TLS) Protocol Version 1.2.*

[IETF RFC 5288] IETF RFC 5288 (2008), *AES Galois Counter Mode (GCM) Cipher Suites for TLS.*

[IETF RFC 6066] IETF RFC 6066 (2011), *Transport Layer Security (TLS) Extensions: Extension Definitions.*

[IETF RFC 5280] IETF RFC 5280 (2008), *Internet X.509 Public Key Infrastructure certificate and certificate Revocation List (CRL) Profile.*

[IETF RFC 6818] IETF RFC 6818 (2013), *Updates to the Internet X.509 Public Key Infrastructure certificate and certificate Revocation List (CRL) Profile.*

[IETF RFC 8446] IETF RFC 8446 (2018), *The Transport Layer Security (TLS) Protocol Version 1.3*.

[W3C PNG] Recommandation W3C (2003), *Portable Network Graphics (PNG) Specification (Second Edition).*

[IETF RFC 6151] IETF RFC 6151 (2011), *Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms.*

[IETF RFC 6125] IETF RFC 6125 (2011), *Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) certificates in the Context of Transport Layer Security (TLS).*

[ISO/CEI 8859-1] ISO/CEI 8859-1:1998, *Technologies de l'information – Jeux de caractères graphiques codés sur un seul octet de 8 bits – Partie 1: Alphabet latin n° 1.*

[ISO 3166-1] ISO 3166-1:2006, *Codes pour la représentation des noms de pays et de leurs subdivisions – Partie 1: codes de pays.*

[ISO 639-2] ISO 639-2:1998, *Codes pour la représentation des noms de langue – Partie 2: code alpha-3.*

[ISO/CEI 62766-5-2] ISO/CEI 62766-5-2:2017, *Consumer terminal function for access to IPTV and open multimedia services - Part 5-2: Web standards TV profile.*

[W3C GIF V89a] W3C, *Graphics Interchange Format version 89a.*

[ISO/CEI 7816-4] ISO/CEI 7816-4, *Cartes d'identification – Cartes à circuit intégré – Partie 4: Organisation, sécurité et commandes pour les échanges*.

3 Définitions

3.1 Termes définis ailleurs

Aucun.

3.2 Termes définis dans la présente Recommandation

La présente Recommandation définit les termes suivants:

Dans la présente Recommandation, l'usage du gras et d'une majuscule initiale signale des termes dont la définition est propre à l'interface ECI et susceptible de différer de leur signification habituelle.

3.2.1 Système de sécurité évoluée: fonction d'un équipement **CPE** conforme **ECI** qui fournit des fonctions de sécurité évoluée (matérielles et logicielles) pour un **Client ECI**.

3.2.2 Créneau de sécurité évoluée: ressources du bloc de sécurité évoluée fournies exclusivement à un **Client ECI** par l'**Hôte ECI**.

3.2.3 Session de créneau de sécurité évoluée: ressources et calcul d'un créneau de **Sécurité évoluée** relatifs au déchiffrement ou au rechiffrement d'un élément de contenu.

3.2.4 Frère: autre **Enfant** du même **Père**.

NOTE – Les termes **Père**, **Enfants** et **Frère** font référence aux entités qui gèrent les **Certificats**.

3.2.5 Certificat: structure de données, telle que définie dans le § 5 de la présente Recommandation, dotée d'une signature numérique sécurisée complémentaire identifiant une **Entité**.

NOTE – Le détenteur de la clé secrète de la signature atteste l'exactitude des données (il les authentifie) en signant avec cette clé. Sa clé publique peut servir à vérifier les données.

3.2.6 Chaîne de certificats: liste de **Certificats** s'authentifiant mutuellement jusqu'à une liste de révocation de racine (incluse).

3.2.7 Sous-système de traitement des Certificats (CPS): sous-système de l'**Hôte ECI** procédant à la vérification des **Certificats** et renforçant la protection contre les altérations volontaires.

3.2.8 Enfant, **Enfants**: **Entité(s)** à laquelle/auxquelles fait référence un **Certificat** signé par un **Père** (commun).

NOTE – Les termes **Père**, **Enfants** et **Frère** désignent les entités qui gèrent les **Certificats**: données d'initialisation et logiciel servant à démarrer le système sur puce d'un **Équipement CPE**.

3.2.9 Système de protection des contenus: système d'un **Écosystème ECI** employant des techniques cryptographiques pour gérer l'accès aux contenus et aux services.

NOTE – Ce terme est souvent interchangeable avec celui de Système de protection des services. Il s'agit habituellement de systèmes d'accès conditionnel (CAS) ou de gestion des droits numériques (DRM).

3.2.10 équipement des locaux d'abonnés (CPE): récepteur de médias ayant mis en œuvre l'interface **ECI** et autorisant l'accès de l'**Utilisateur** à des services de médias numériques.

3.2.11 Fabricant d'équipement CPE: société qui produit des **Équipements CPE** conformes **ECI**.

3.2.12 Interface ECI (interface commune intégrée): architecture et système spécifiés dans le cadre du groupe ETSI ISG "Embedded CI", qui permettent de créer et de mettre en œuvre des **Clients ECI** interchangeables dans l'équipement des locaux d'abonnés (**CPE**) et assurent ainsi l'interopérabilité des dispositifs **CPE** en ce qui concerne l'interface **ECI**.

3.2.13 Application ECI: application HTML hébergée sur un **Client ECI** et s'exécutant lors d'une session de navigateur dédiée dans le but d'interagir avec l'**Utilisateur** et de lui permettre d'entrer des données au niveau du **Client ECI**.

3.2.14 Fabricant de puce ECI: société fournissant des systèmes sur puce mettant en œuvre des fonctionnalités de puce spécifiées **ECI**.

3.2.15 Client ECI (client d'une interface commune intégrée): mise en œuvre d'un client CA/DRM qui est conforme aux spécifications d'interface commune intégrée.

NOTE – Il s'agit du module logiciel d'un **Équipement CPE** qui met à disposition l'ensemble des moyens permettant de recevoir, de manière protégée, les habilitations et les droits d'un consommateur relatifs aux contenus distribués par un distributeur de contenus ou un **Opérateur** et d'en commander l'exécution. Le module reçoit en outre les conditions selon lesquelles un droit ou une habilitation peut être utilisé(e) par le consommateur, et les clés permettant de déchiffrer les différents messages et contenus.

3.2.16 Image de client ECI: fichier contenant un code logiciel de machine virtuelle et les données d'initialisation requises par le **chargeur de Client ECI**.

3.2.17 Chargeur de Client ECI: partie du module logiciel de l'**Hôte ECI** permettant de télécharger, de vérifier et d'installer un nouveau logiciel **Client ECI** dans un **Conteneur ECI** de l'**Hôte ECI**.

3.2.18 Conteneur ECI: instance unique de machine virtuelle avec bibliothèques de support complémentaires et API **ECI** permettant à une instance unique de **Client ECI** de s'exécuter sur un **Équipement CPE**.

3.2.19 Écosystème ECI: opération commerciale consistant en une **Autorité de confiance** et plusieurs plates-formes et **Équipements CPE** conformes **ECI** sur le terrain.

3.2.20 Hôte ECI: système matériel et logiciel d'un **Équipement CPE**, qui couvre les fonctionnalités liées à l'interface **ECI** et comporte des interfaces vers un **Client ECI**.

NOTE – L'**Hôte ECI** fait partie du micrologiciel de l'**Équipement CPE**.

3.2.21 Image d'hôte ECI: fichier(s) contenant un logiciel et des données d'initialisation pour un environnement **ECI**.

NOTE 1 – Une image d'**Hôte ECI** peut comprendre plusieurs fichiers d'**images d'Hôte ECI**.

NOTE 2 – Elle peut également contenir un autre logiciel qui n'interfère pas avec l'**Hôte ECI** ou n'en autorise pas l'observation inappropriée.

3.2.22 chargeur de l'Hôte ECI: module logiciel qui permet de télécharger, de vérifier et d'installer un logiciel d'**Hôte ECI** dans un équipement **CPE**.

NOTE – Dans une configuration de chargement en plusieurs étapes, ce terme désigne toutes les fonctions de chargement essentielles à la sécurité associées au chargement de l'Hôte ECI.

3.2.23 Certificat racine ECI: **Certificat** émis pour vérifier les éléments approuvés par une **Autorité de confiance ECI**.

3.2.24 Entité: organisation (par exemple, fabricant, **Opérateur** ou **Fournisseur de systèmes de sécurité**) ou dispositif physique (par exemple., **Hôte ECI**, **Opération de plate-forme** ou **Client ECI**) désigné(e) par un identificateur unique dans un **Écosystème ECI**.

3.2.25 Chaîne d'exportation: chaîne de **Certificats** servant à autoriser l'exportation d'un ou plusieurs **systèmes Micro DRM**.

3.2.26 Connexion d'exportation: relation authentifiée entre un **Client ECI** capable de déchiffrer un contenu et un **Micro serveur** capable de le rechiffrer.

3.2.27 Groupe d'exportation: groupe de **Micro systèmes DRM** vers lequel l'exportation est autorisée.

3.2.28 Père: signataire du **Certificat** de l'**Entité enfant**.

NOTE – Les termes **Père, Enfants** et **Frère** désignent les entités qui gèrent des **Certificats**.

3.2.29 Série d'images: série d'images d'un **Hôte ECI** ou d'un **Client ECI** différentes en fonction de l'identificateur (**CPE**_id) de l'**Équipement CPE** mais dont les fonctionnalités sont (presque) identiques.

3.2.30 Chaîne d'importation: chaîne allant de la Clé publique d'opération de plate-forme d'un **Client ECI** à une **Entité** représentant un système d'exportation ou un **Groupe d'exportation**.

NOTE – Il est possible d'utiliser une **Chaîne d'exportation** et la **Chaîne d'importation** correspondante pour authentifier une session de **Micro serveur** important des contenus à destination d'un **Client ECI** exportateur.

3.2.31 Connexion d'importation: connexion approuvée entre un **Client ECI** et un **Micro serveur** lui permettant d'importer des contenus déchiffrés à des fins de rechiffrement ultérieur.

3.2.32 Fabricant: entité développant et vendant des équipements CPE prenant en charge une mise en œuvre du système **ECI** et permettant d'installer des **Hôte**s **ECI** et des **Clients ECI** via le téléchargement d'un logiciel.

3.2.33 Pointeur de média: référence à une configuration de traitement de déchiffrement ou de rechiffrement d'un seul programme entre un **Client ECI** et un **Hôte ECI.**

3.2.34 Micro client: Client ECI ou autre capable de déchiffrer des contenus rechiffrés par un **Micro serveur**.

3.2.35 Micro serveur: **Client ECI** capable d'importer des contenus déchiffrés, de les rechiffrer et d'authentifier un **Client ECI** ou un groupe de **Clients ECI** spécifique en tant que **Cible** d'un déchiffrement ultérieur.

3.2.36 Système Micro DRM: **Système de protection des contenus** rechiffrant les contenus sur un **Équipement CPE** avec un **Micro serveur** et permettant le décodage de ces contenus rechiffrés par des **Micro clients** authentifiés.

NOTE – Le **Micro serveur** et les **Micro client**s sont fournis par un opérateur de **Système Micro DRM**.

3.2.37 Opérateur: organisation fournissant des **opérations de plate-forme**, enregistrée auprès de l'**Autorité de confiance ECI** en tant que signataire de l'**Écosystème ECI**.

NOTE – Un **Opérateur** peut effectuer de multiples **opérations de plate-forme**.

3.2.38 Opération de plate-forme: instance particulière d'une opération de prestation de services techniques dotée d'une seule identité **ECI** en matière de sécurité.

3.2.39 Session de rechiffrement: processus commandé par un **Micro Serveur** consistant à importer des contenus à partir d'une **Connexion d'importation**, de les rechiffrer et de produire les informations nécessaires au déchiffrement ultérieur de ces contenus par la **Cible** authentifiée.

3.2.40 Requête: message d'un émetteur demandant à un récepteur certaines informations ou l'exécution de certaines opérations au sein d'un **Écosystème ECI**, spécifiées dans les champs de données de la requête en question.

NOTE – Le paragraphe 9.2.3 fournit des détails complémentaires.

3.2.41 Réponse: message émis dans un **Écosystème ECI** pour répondre à une **Requête.**

NOTE – Le paragraphe 9.2.3 fournit des détails complémentaires.

3.2.42 Liste de révocation: liste de **Certificats** révoqués et, de ce fait, désormais inutilisables.

3.2.43 Racine: clé publique ou **Certificat** contenant une clé publique servant de base à l'authentification d'une chaîne de **Certificats**.

3.2.44 Canal authentifié sécurisé: trajet de communication (canal) établi entre deux **Entités** où celles-ci se sont identifiées mutuellement de façon sécurisée (authentification) et ont convenu de chiffrer les données transférées de l'une à l'autre (sécurisation).

3.2.45 Service: contenu fourni par une **Opération de plate-forme**.

NOTE – Dans le contexte de l'interface **ECI**, seuls les contenus protégés sont pris en compte.

3.2.46 Clé publique d'émetteur (SPK): clé publique de l'émetteur des contenus chiffrés utilisée dans un **Écosystème ECI** pour vérifier l'origine de la signature de la première clé d'une chaîne de clés servant à déchiffrer les contenus, l'émetteur prenant part à une **Opération de plate-forme**.

3.2.47 Carte à puce: dispositif de sécurité matériel séparable utilisé par plusieurs fournisseurs de systèmes CA ou DRM pour renforcer le niveau de sécurité de leurs produits dans un **Écosystème ECI**.

3.2.48 Cible: **Micro client** ou groupe de **Micro Client**s dont les contenus sont rechiffrés par un **Micro serveur**.

3.2.49 Autorité de confiance: organisation régissant l'ensemble des règles et des règlements s'appliquant à une mise en œuvre donnée d'une interface **ECI** et ciblant un certain marché.

NOTE – L'**Autorité de confiance** doit être une entité juridique pour pouvoir régler les réclamations fondées en droit. Elle doit faire preuve d'impartialité envers tous les acteurs de l'**Écosystème ECI** qu'elle régit.

3.2.50 Tiers de confiance: fournisseur de services de sécurité délivrant des **Certificat**s et des clés aux **Fabricant**s conformes des composants pertinents d'un **système ECI**.

NOTE – Il est placé sous le contrôle de l'**Autorité de confiance**.

3.2.51 Utilisateur: personne se servant d'un dispositif conforme **ECI**.

3.2.52 Instance de Machine virtuelle: instanciation de machine virtuelle établie par un **Hôte ECI** se présentant à un **Client ECI** comme un environnement d'exécution.

4 Abréviations et acronymes

La présente Recommandation utilise les abréviations et acronymes suivants:

4CC	Code à quatre caractères (*également dénommé FourCC*)
3DES	Triple algorithme DES
AEAD	Chiffrement authentifié avec données associées (*Authenticated Encryption with Associated Data*)
AES	Norme de chiffrement perfectionné (*Advanced Encryption Standard*)
AES-GCM	Mode compteur AES Galois (*AES Galois Counter Mode*)
AID	Identificateur d'application (*Application IDentifier*)
AK	Clé d'authentification (*Authentification Key*)
APDU	Unité de données de protocole d'application (*Application Protocol Data Unit*)
API	Interface de programmation d'application (*Application Programming Interface*)
AS	Sécurité évoluée (*Advanced Security*)
ASCII	American Standard Code for Information Interchange
ATR	Réponse de réinitialisation (*Answer to Reset*)
BAT	Table d'association de bouquet (*Bouquet Association Table*)
BMFF	Format de base pour les fichiers médias (*Base Media File Format*)
BSD	Berkeley Software Distribution
CA	Accès conditionnel (*Conditional Access*)
CA/DRM	Accès conditionnel/Gestion des droits numériques (*Conditional Access/Digital Rights Management*)
CAT	Table d'accès conditionnel (*Conditional Access Table*)
CBC	Mode chaînage de blocs chiffrants (*Cipher Block Chaining*)
CENC	Chiffrement commun (*Common Encryption*)
CI	Interface commune (*Common Interface*)
CP	Propriété de contenu (*Content Property*)
CPE	équipement des locaux d'abonnés (*Customer Premises Equipment*)
CPS	Sous-système de traitement des certificats (*certificate Processing Subsystem*)
CPU	Unité centrale de traitement (*Central Processing Unit*)
CRC	Contrôle de redondance cyclique (*Cyclic Redundancy Check*)
CRL	liste de révocation des certificats (*certificate Revocation List*)

CSA	Algorithme de brouillage commun (*Common Scrambling Algorithm*)
CSA1	Algorithme de brouillage commun, première version
CSA3	Algorithme de brouillage commun, troisième version
CSS	Feuilles de style en cascade W3C (*W3C Cascading Style Sheets*)
CSS3	CSS version 3
CTR	Mode compteur (*Counter Mode*)
CW	Mot de contrôle (*Control Word*)
Dash	Streaming adaptatif dynamique sur HTTP (*Dynamic Adaptive Streaming over HTTP*)
DDB	Bloc de données à télécharger (*Download Data Block*)
DDOS	Déni de service réparti (*Distributed Denial of Service*)
DES	Norme de chiffrement des données (*Data Encryption Standard*)
DHE	Mode éphémère Diffie-Hellman (*Ephemeral Diffie-Hellman*)
DII	Indication d'informations relatives au téléchargement (*Download Info Indication*)
DLNA	Digital Living Network Alliance
DNS	Système de noms de domaine (*Domain Name System*)
DRM	Gestion des droits numériques (*Digital Rights Management*)
DSI	Lancement du serveur de téléchargement (*Download Server Initiate*)
DSMCC	Commande et contrôle de supports d'enregistrement numérique (*Digital Storage Media Command and Control*)
DVB	Radiodiffusion vidéonumérique (*Digital Video Broadcasting*)
EAC	certificat d'autorisation d'exportation (*Export Authorization certificate*)
EAOC	certificat d'opérateur d'autorisation d'exportation (*Export Authorization Operator certificate*)
ECM	Message de commande d'habilitation (*Entitlement Control Message*)
EGC	certificat de groupe d'exportation (*Export Group certificate*)
EIT	Table d'informations d'événement (*Event Information Table*)
EMM	Message de gestion d'habilitation (*Entitlement Management Message*)
ES	Flux élémentaire (*Elementary Stream*)
ESC	certificat de système d'exportation (*Export System certificate*)
GCM	Mode compteur/Galois (*Galois/Counter Mode*)
GMT	Heure moyenne de Greenwich (*Greenwich Mean Time*)
HD	Haute définition (*High Definition*)
HDCP	Protection des contenus numériques à grande largeur de bande (*High-bandwidth Digital Content Protection*)
HTML	Langage de balisage hypertexte (*Hyper Text Mark-up Language*)
HTTP	Protocole de transfert hypertexte (*Hypertext Transfer Protocol*)
HTTP(S)	Protocole de transfert hypertexte sécurisé (*Hypertext Transfer Protocol Secure*)

iDTV	Récepteur de télévision numérique intégré (*integrated Digital TV receiver*)
IFSC	Taille de champ d'information de carte (*Information Field Size of Card*)
IFSD	Taille de champ d'information de dispositif (*Information Field Size of Device*)
IP	Protocole Internet (*Internet Protocol*)
IPv4	Protocole Internet version 4
IPv6	Protocole Internet version 6
ISO	Organisation internationale de normalisation (*International Organization for Standardisation*)
ISOBMFF	Format ISO de base pour les fichiers médias (*ISO Base Media File Format*)
LAN	Réseau local (*Local Area Network*)
LSB	Bit de plus faible poids (*Least Significant Bit*)
MIME	Extensions de courrier Internet à fonctions multiples (*Multipurpose Internet Mail Extensions*)
MMI	Interface homme-machine (*Man Machine Interface*)
MP4	Format de conteneur multimédia numérique (*Digital Multimedia Container Format*) (appelé également MPEG-4 partie 14)
MPD	Description de la présentation des médias
MPEG	Groupe d'experts pour les images animées (*Motion Picture Experts Group*)
MSB	Bit de poids le plus fort (*Most Significant Bit*)
NV memory	Mémoire non volatile (*Non-Volatile memory*)
NV	Non volatile (*Non-Volatile*)
OS	Système d'exploitation (*Operating System*)
OTT	Fourniture de services audio et vidéo par Internet en utilisant les structures existantes installées par un autre acteur (*over the top*)
OUI	Identificateur propre à une organisation (*Organizationnally Unique Identifier*)
PAT	Table d'association de programme (*Program Association Table*)
PayTV	Télévision à péage
PES	Flux élémentaire de paquets (*Packet Elementary Stream*)
PID	Identificateur de paquet MPEG (*MPEG Packet Identifier*)
PIN	Numéro personnel d'identification (*Personal Identification Number*)
PKIX	Infrastructure de clé publique X.509 (*Public-Key Infrastructure X.509*)
PMT	Table de mappage des programmes (*Program Map Table*)
PO	Opération de plate-forme (*Platform Operation*)
POC	certificat d'opération de plate-forme (*Platform Operation certificate*)
POPK	Clé publique d'opération de plate-forme (*Platform Operation Public Key*)
PPS	Sélection du protocole et des paramètres (*Protocol and Parameter Selection*)
PSI	Informations propres aux programmes (*Program Specific Information*)
PSSH	En-tête propre au système de protection (*Protection System Specific Header*)

PVR	Enregistreur vidéo personnel (*Personal Video Recorder*)
RAM	Mémoire vive (*Random Access Memory*)
RFU	Réservé à une utilisation future (*Reserved for Future Use*)
RL	liste de révocation (*Revocation List*)
s. o.	sans objet
SAC	Canal authentifié sécurisé (*Secure Authenticated Channel*)
SDT	Table de description de service (*Service Description Table*)
SHA	Algorithme de hachage sécurisé (*Secure Hash Algorithm*)
SI	Information relative au service (*Service Information*)
SIM	Module d'identification de l'abonné (*Subscriber Identity Module*)
SoC	Système sur puce (*System on Chip*)
SPK	Clé publique de signature (*Signature Public Key*), également appelée Clé de vérification de signature (*Signature Verification Key*)
SSK	Clé secrète de signature (*Signature Secret Key*), également appelée Clé privée de signature (*Signature Private Key*)
SSL	Couche de connecteurs sécurisés (*Secure Sockets Layer*)
SSU	Mise à jour des logiciels du système (*System Software Update*)
STB	Décodeur (*Set Top Box*)
TA	autorité de confiance (*Trust Authority*)
TCK	Octet de vérification (*Check byte*)
TCP	Protocole de commande de transmission (*Transmission Control Protocol*)
TLS	Sécurité de la couche de transport (*Transport Layer Security*)
TPC	Protocole de puissance de transmission (*Transmission Power Protocol*)
TPDU	Unité de données de protocole de transport (*Transport Protocol Data Unit*)
TPEGC	certificat de groupe d'exportation tiers (*Third Party Export Group certificate*)
TS	Flux de transport (*Transport Stream*)
TTP	Tiers de confiance (*Trusted Third Party*)
TV	Télévision
TVIP	Téléviseur utilisant le protocole Internet (IP)
UDP	Protocole de datagramme utilisateur (*User Datagram Protocol*)
UHD	Ultra haute définition (*Ultra High Definition*)
UI	Interface d'utilisateur (*User Interface*)
uimsbf	Entier non signé, bit de plus fort poids en premier (*unsigned integer, most significant bit first*)
UNT	Table de notification des mises à jour (*Update Notification Table*)
URI	Informations relatives aux droits d'utilisation (*Usage Rights Information*)
URL	Localisateur uniforme de ressource (*Uniform Resource Locator*)

USB	Bus série universel (*Universal Serial Bus*)
UTF	Format de codage UCS (*jeu de caractères universel*)
UUID	Identificateur unique universel (*Universally Unique Identifier*)
VM	machine virtuelle (*Virtual Machine*)
WAN	Réseau étendu (*Wide Area Network*)
WEB	World Wide Web

5 Système de Certificats ECI

5.1 Introduction

5.1.1 Domaine d'application

L'interface **ECI** utilise des **Certificats** à diverses fins, tels que **Certificats de Chargeur d'hôte ECI, de chargeur de Client ECI** et **d'Opérateur ECI**. Le présent paragraphe définit ces **Certificats** et la **Liste de révocation** qui y est associée, ainsi que leur organisation en chaînes et la structure du **Certificat racine**. Cette définition utilise un format binaire compact spécifié dans la présente Recommandation, adaptable aux mises en œuvre matérielles et convenant à la cryptographie, ainsi qu'un système de signalisation simple en prévision des versions et des extensions futures.

5.1.2 Notation et conventions adoptées pour les champs

Les définitions de la structure des données ci-après sont directement associées à une séquence d'octets. Les fonctions cryptographiques sont définies de manière à s'exécuter sur la représentation d'une séquence d'octets.

La définition des données suit un alignement naturel pour les champs de 16 octets et 32 octets afin de simplifier leur traitement sur un processeur central 32 bits. Un bourrage sert de champ générique pour indiquer les champs de remplissage requis à cette fin. Il utilise la fonction padding(n_bytes) où n_bytes est la frontière d'alignement en nombre d'octets à partir du début de la structure de données définie. L'interprétation des structures de données n'abordera pas les champs de bourrage. La valeur du champ de bourrage sera mise à 0.

Les champs définis par une autre structure de données à l'aide d'une définition de type n'ont pas de mnémonique. En général, il ne leur est pas attribué de définition de longueur de champ.

5.1.3 Champ d'extension

Un grand nombre des structures de données plus importantes définies possèdent un champ d'extension autorisant l'ajout futur d'extensions (compatibles avec les extensions antérieures). La définition est donnée dans le Tableau 5.1.3-1.

Tableau 5.1.3-1 – Définition des champs d'extension

Syntaxe	Nbre de bits	Mnémonique
`Extension_Field {`		
`padding(4)`		
`length`	32	`uimsbf`
`for (i=0; i<length; i++) {`		
`extension_byte`	8	`uimsbf`
`}`		
`}`		

Sémantique:

length: nombre entier	Nombre d'octets dans la boucle suivante. La valeur est en principe un multiple de 4 et peut être 0.
extension_byte: octet	Champ de données contenant des informations que peuvent ignorer les mises en œuvre basées sur des versions du présent document n'ayant pas défini le contenu de ce champ.

5.2 Certificats ECI

Le **Certificat ECI** possède une structure simple. Son identificateur est tout simplement un nombre binaire réservé aux interprétations machine, contrairement aux certificats X.509 utilisés sur Internet.

Sa structure générique est illustrée dans la Figure 5.2-1.

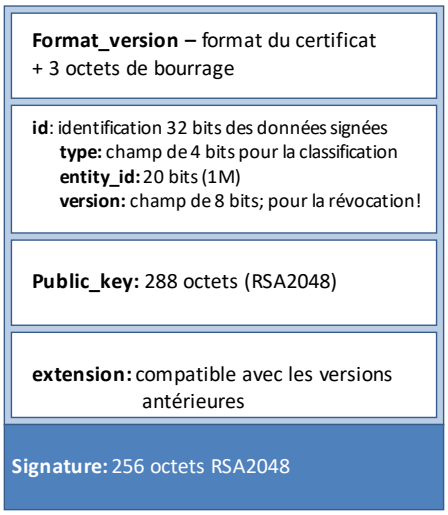


Figure 5.2-1 – Format de Certificat ECI, version 1

Le format des **Certificat**s **ECI** est défini dans le Tableau 5.2-1.

Chaque élément signé utilisera un champ de départ distinct de 8 octets où le premier octet correspond au format de la version de l'élément signé. Il est suivi (pour les éléments de la version 1) par 3 octets de bourrage, les 4 octets suivants représentant un identificateur unique dans le contexte de la clé secrète de l'**Entité** signataire.

Tableau 5.2-1 – Définition des Certificats ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_certificate_Id {		
padding(4)		
Type	4	uimsbf
entity_id	20	uimsbf
Version	8	uimsbf
}		
ECI_Public_Key_v1 {		
byte **modulus**[256]	2 048	
}		
ECI_certificate_Data_v1 {		
ECI_certificate_Id **id**	32	uimsbf
Public_Key_v1 **public_key**	2 304	
Extension_Field **extension**		
}		
ECI_Signature_v1 {		
byte **signature**[256]	2 048	uimsbf
}		
ECI_certificate_Id {		
format_version	8	uimsbf
if (version == 0x01) {		
ECI_certificate_Data_v1 **data**		
ECI_Signature_v1 **signature**		
}		
}		

Sémantique:

format_version: nombre entier	Valeurs 0x00, 0x02..0xFF: réservées. Valeur 0x01: format de **Certificat ECI,** version 1. Les mises en œuvre qui ne reconnaissent pas un type de **Certificat** ne le traitent pas et les requêtes de validation renvoient une indication d'échec.
id: nombre entier	Identification du certificat sous la forme d'un nombre de 32 bits unique dans le contexte du **Père du Certificat** (signataire du **Certificat**). Les valeurs 0x00000 et 0xF0000-0xFFFFF sont réservées.
type: nombre entier	**type** définit le type de l'Entité, par exemple **Fabricant**, **Hôte ECI**, **Opérateur**, etc., dans le contexte du signataire (**Père**). Les **Certificats** dotés d'une valeur de type 0x0 .. 0x7 nécessiteront une **Liste de révocation** pour la vérification des **Enfants**. Les valeurs de type 0x8 et au-delà ne nécessiteront pas de **Liste de révocation** pour la vérification des **Enfants** (voir le Tableau 5.2-2).
entity_id: nombre entier	Définit le numéro de l'**Entité**. Le type de **Certificat** définit les divers sous-formats de **entity_id**. Sauf indication contraire, les identificateurs d'Entités sont uniques dans le contexte du **Père** (signataire du **Certificat** ou de la **Liste de révocation**).
version:	Numéro de version du **Certificat** de l'entité attribué par ordre croissant (en général, augmentant de 1).
extension: Extension_Field	Les fonctions de traitement non définies pour interpréter les données de ce champ les ignoreront. Ce champ peut être utilisé pour certaines données en cas d'application particulière de la définition générique du **Certificat**. Son interprétation est fonction du contexte. Ce champ ne servira pas pour des applications non **ECI**, sauf autorisation explicite.
public_key: ECI_Public_Key_v1	Clé publique (attribuée par le **Père**) de l'Entité de ce **Certificat**.
data: ECI_certificate_Data	Il s'agit de la section du **Certificat** contenant des données.
signature: octet[256]	Le champ de signature contient la représentation de la séquence d'octets de la signature du **Père** du **Certificat** utilisant les fonctions cryptographiques telles que définies dans l'Annexe A.

Toute vérification d'un **Certificat ECI** portera notamment sur sa longueur totale en termes d'accumulation de définitions des champs.

Des valeurs génériques types sont utilisées pour la plupart des **Certificats** et des **Listes de révocation** afin que toutes les valeurs attribuées soient uniques. Le Tableau 5.2-2 donne une vue d'ensemble de toutes les données signées par l'**Autorité de confiance ECI**.

Tableau 5.2-2 – Attribution des identificateurs et Pères des éléments signés

Père	Type	Champ d'identification	Description
Racine	0x0	0xFFFFF	Racine
Racine	0x1	Identificateur de Fabricant, <> 0xFxxxx	**Certificat** de Fabricant
Racine	0x1	Identificateur de liste de révocation de Fabricant, <> 0xFxxxx	**Liste de révocation** de Fabricant
Fabricant	0x0	Host_id, <> 0xFxxxx	**Certificat d'Hôte ECI**
Fabricant	0x0	Host RL, == 0xFxxxx	**Liste de révocation d'Hôte ECI**
Hôte	0x8	Identificateur d'image d'hôte	**Image de l'Hôte ECI**
Hôte	0x9	Identificateur de série d'images d'Hôte	**Certificat de série d'images d'Hôte ECI**
Série d'images d'Hôte	0x9	Identificateur de Cible d'image	Image de série d'**Hôte ECI**
Racine	0x2	Identificateur de fournisseur, <> 0xFxxxx	**Certificat de fournisseur de systèmes de sécurité**
Racine	0x2	Identificateur de liste de révocation de fournisseur, == 0xFxxxx	**Liste de révocation de fournisseur de systèmes de sécurité**
Fournisseur	0x0	Identificateur du client, <> 0xFxxxx	**Certificat de Client ECI**
Fournisseur	0x0	liste de révocation de client, == 0xFxxxx	**Liste de révocation** de **Clients ECI** et de séries de **Clients ECI**
Client	0x0	Identificateur de Client	**Image de client ECI**
Client	0x1	Identificateur de série de Clients	**Certificat** de série de Clients
Série de Clients	0x8	Identificateur de Cible d'image	Image de série de Clients
Racine	0x3	Identificateur d'opérateur, <> 0xFxxxx	**Certificat** d'opérateur
Racine	0x3	Identificateur de liste de révocation d'opérateur, == 0xFxxxx	**Liste de révocation** d'opérateur
opérateur	0x0	Identificateur d'Opération de plate-forme,<> 0xFxxxx	**Certificat d'opération de plate-forme**
opérateur	0x0	Identificateur de liste de révocation d'Opération de plate-forme, == 0xFxxxx	liste de révocation d'**Opération de plate-forme**
Opération de plate-forme	0x0	Identificateur de cosignature d'image de client d'Opération de plate-forme, <> 0xFxxxx	Cosignature d'image de client d'**Opération de plate-forme**
Opération de plate-forme	0x0	Identificateur de liste de révocation "image de client d'Opération de plate-forme, == 0xFxxxx	**Liste de révocation d'image** de Client d'**Opération de plate-forme**
Opération de plate-forme ou Groupe cible	0x0	Identificateur de Groupe cible, <> 0xFxxxx	Groupe cible défini dans la Recommandation [UIT-T J.1014]
Opération de plate-forme ou Groupe cible	0x0	Identificateur de liste de révocation de cible, == 0xFxxxx	liste de révocation de cible définie dans la Recommandation [UIT-T J.1014]
Opération de plate-forme ou Groupe cible	0x8	Identificateur de Micro client, <> 0xFxxxx	Micro client, défini dans la Recommandation [UIT-T J.1014]
Opération de plate-forme, **Groupe d'exportation**, Groupe d'exportation tiers	0x4	Identificateur de Groupe d'exportation, <> 0xFxxxx	**Groupe d'exportation**
Opération de plate-forme, **Groupe d'exportation**, Groupe d'exportation tiers	0x4	Identificateur de liste de révocation de Groupe d'exportation, == 0xFxxxx	liste de révocation de **Groupe d'exportation**

Père	Type	Champ d'identification	Description
Groupe d'exportation	0x5	Identificateur de Groupe d'exportation tiers, <> 0xFxxxx	**Groupe d'exportation** tiers
Groupe d'exportation	0x8	Identificateur de liste de révocation de Groupe d'exportation, == 0xFxxxx	liste de révocation de **Groupe d'exportation**
Groupe d'exportation Groupe d'exportation tiers	0xE	Identificateur de Système d'exportation, <> 0xFxxxx	Système d'exportation
Racine	0x4	Identificateur d'opérateur d'autorisation d'exportation, <> 0xFxxxx	opérateur d'autorisation d'exportation
Racine	0x4	Identificateur d'opérateur d'autorisation d'exportation, == 0xFxxxx	liste de révocation d'opérateur d'autorisation d'exportation
opérateur d'autorisation d'exportation, Autorisation d'exportation	0x0	Identificateur d'autorisation d'exportation, <> 0xFxxxx	Autorisation d'exportation (avec **Enfants**)
opérateur d'autorisation d'exportation, Autorisation d'exportation	0x0	Identificateur d'autorisation d'exportation, == 0xFxxxx	liste de révocation d'autorisation d'exportation
Autres	Autres		Réservé
NOTE – Les fonctions ECI peuvent transporter et traiter le champ **data** et les sections **signature** d'un **Certificat** ou un autre élément de données signé séparément.			

5.3 Listes de révocation ECI

La **Liste de révocation** est signée par l'**Entité** signataire à l'origine du **Certificat** révoqué. Elle contient la liste des identificateurs des entités définissant la version acceptable minimale de leurs **Certificats**. Si une liste de révocation comporte un **Certificat** auquel une ou plusieurs listes de révocation sont associées, il convient d'appliquer à ce **Certificat** un numéro minimum de version de **Liste de révocation**. La structure des **Listes de révocation ECI** est présentée dans la Figure 5.3-1.

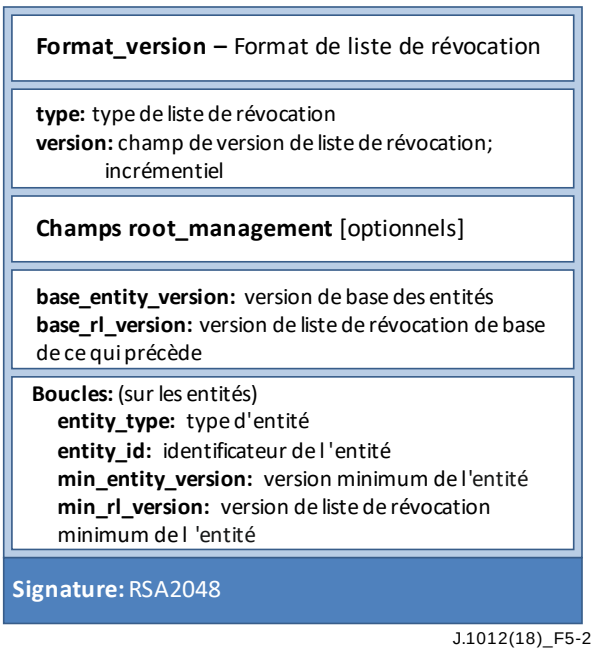


Figure 5.3-1 – Structure des Listes de révocation

Les mises en œuvre de l'**Hôte ECI** stockeront la **Liste de révocation** reçue la plus récemment (définie dans le champ **rl_version**) en lien avec une **Entité** qu'elles gèrent, quelle que soit la source des données.

La **Liste de révocation** (ECI_RL) est définie dans le Tableau 5.3-1.

Tableau 5.3-1 – Définition de la Liste de révocation

Syntaxe	Nbre de bits	Mnémonique
ECI_RL_Id {		
padding(4)		
Type	4	uimsbf
indicator = 0xF	4	uimsbf
version:	24	uimsbf
}		
ECI_Revocation_List_v1 {		
base_entity_version	8	uimsbf
base_rl_version	24	uimsbf
number_of_entities	24	uimsbf
for (i=0; i<number_of_entities; i++){		
entity_type	4	uimsbf
entity_id	20	uimsbf
min_entity_version	8	uimsbf
min_rl_version	24	uimsbf
}		
}		
ECI_RL {		
format_version	8	uimsbf
if (format_version == 0x01){		
ECI_RL_Id **rl_id**	32+24	uimsbf
root_version_indicator	1	uimsbf
padding(1)	7	uimsbf
root_version	8	uimsbf
min_root_version	8	uimsbf
padding(4)		
ECI_Revocation_List_v1 **rev_list**		
Extension_Field **extension**		
ECI_Signature_v1 **rl_signature**	2 048 (voir la NOTE)	uimsbf
}		
}		
NOTE = dans les listes de révocation de certificats associées aux **Certificat**s de version 1.		

Sémantique:

format_version: nombre entier	Valeurs 0x00, 0x02..0xFF: réservées. Valeur 0x01: format version 1 de la **Liste de révocation** ECI. Les mises en œuvre qui ne reconnaissent pas un type de **Certificat** ne le traitent pas et les requêtes de validation renvoient une indication d'échec.
type: nombre entier	Le type de champ est défini dans ECI_certificate_Id, voir le Tableau 5.3-1.
indicator: nombre entier	Indication de la **Liste de révocation**. Sa valeur sera égale à 0xF.
version: nombre entier	Version de cette liste de révocation. Commence à 1 (en général vide avec un nouveau **Certificat**) et augmente à chaque mise à jour.
base_entity_version: nombre entier	Toutes les entités dont le champ **id.version** est inférieur au champ **base_id_version** sont révoquées.
base_rl|_version	Toutes les listes de révocation d'une Entité dont la version est égale au champ **base_entity_version** et qui sont inférieures au champ **base_rl_version** ne sont plus valables.
number_of_entities: nombre entier	Nombre d'Entités figurant dans la liste de révocation. Voir le Tableau 5.3-1 ci-dessous pour connaître les valeurs maximales.
entity_type: nombre entier	Type d'Entité dont les anciennes versions sont révoquées.
entity_id: nombre entier	Identificateur de l'Entité dont les anciennes versions sont révoquées.
min_entity_version: nombre entier	Numéro de version minimum de l'Entité (identificateur de certificat) correspondant à **entity_type** et **entity_id**. Les versions inférieures sont révoquées.
min_rl_version	Version minimum de la liste de révocation à appliquer avec une Entité correspondant à **entity_type**, **entity_id** et **entity_min_version**. Les versions inférieures de la liste de révocation ne sont plus valables.
root_version_indicator: bit	Si la valeur est nulle, les champs root_version et min-root_version n'auront pas de signification. Si la valeur est égale à 1 et que le **Père** est un **Certificat racine**, les champs **root_version** et **min_root_version** seront interprétés comme ci-dessous.
root_version	Version du **Certificat racine** signataire de la **Liste de révocation** concernée.
min_root_version: nombre entier	Si la version du **Père** (c.-à-d. de la Racine) est supérieure ou égale à ce champ, toutes les versions du **Certificat racine** inférieures à **min_root_version** seront révoquées pour la vérification des **Certificats** du type défini dans le champ **revocation_id_lead**.
extension: Extension_Field	Données supplémentaires: ce champ sera ignoré par les mises en œuvre non conçues pour l'interpréter, hormis aux fins de calcul de la signature.
rl_signature: ECI_Signature_v1	Signature de l'**Entité ECI** à laquelle la **Liste de révocation** est associée. La signature est calculée à partir de toutes les données précédentes.

NOTE – Les mises en œuvre matérielles peuvent traiter les **Listes de révocation** par fragments et rechercher l'identificateur du **Certificat** suivant à valider lors du cumul de hachage de la signature et de la vérification de la signature en fin de liste.

En général, les **Hôtes ECI** stockeront les **Listes de révocation de l'Autorité de confiance** de tous les **Certificats** requis pour vérifier les entités chargées par l'**Hôte ECI**. Les **Hôtes ECI** remplaceront la **Liste de révocation** stockée d'un **Certificat** ou d'un élément par une **Liste de révocation** nouvellement reçue et dont le numéro de version est plus récent.

La longueur maximale des **Listes de révocation** sera conforme au § B.2.

5.4 Chaînes de Certificats et Arborescences de Listes de révocation

5.4.1 Définitions de la structure de données

Une **Chaîne de certificats** est une séquence de **Certificats** associés à des **Listes de révocation** dont chaque **Certificat** a été signé par l'entité gestionnaire du **Certificat** précédent. Elle commence par la **Liste de révocation** du **Certificat de Père** (en général une Racine). Le numéro de version (valable) minimum du **Certificat** et la version (valable) minimum de la **Liste de révocation** d'un **Enfant** sont définis par la **Liste de révocation** de son **Père**. Étant donné que les chaînes servent de justificatifs d'identité visant à vérifier les éléments à charger, un **Certificat** ne figure pas, en général, sur la **Liste de révocation** de son prédécesseur. Néanmoins, le traitement de la **Liste de révocation** est

obligatoire afin de vérifier l'intégrité de la chaîne. Le Tableau 5.4.1-1 présente la structure d'une **Chaîne de certificats** type.

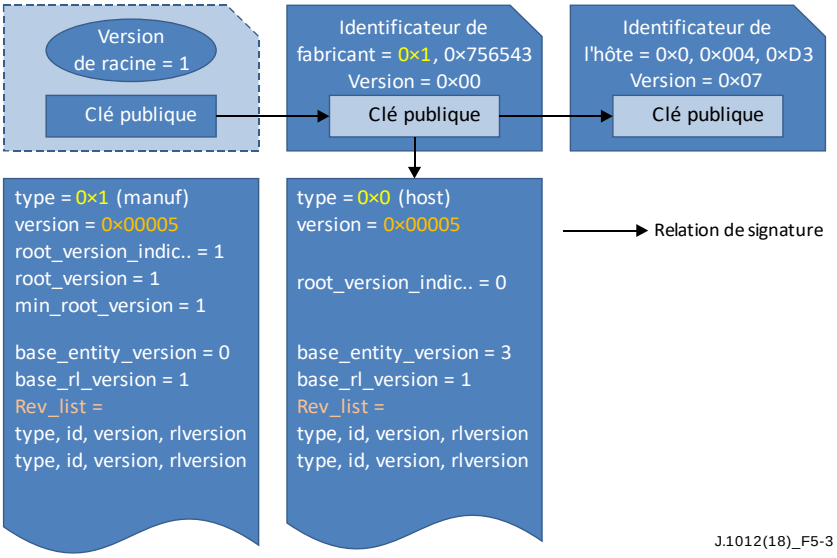


Figure 5.4.1-1 – Exemple de Chaîne de certificats d'hôte

Les chaînes peuvent être transportées ou stockées et peuvent comporter différentes sections.

Les arborescences de listes de révocation sont des séquences de listes de révocation liées entre elles qui utilisent comme **Père** un **Certificat** d'une chaîne précédente et couvrent donc de nombreux éléments certifiés. Les **opérations de plate-forme** peuvent s'en servir pour désapprouver (indiquer la révocation) d'autres entités (révoquées par l'**Autorité de confiance**). La définition des **Chaînes de Certificats** et des Arborescences de **listes de révocation** sera conforme au Tableau 5.4.1-1.

Tableau 5.4.1-1 – Définition des Chaînes de Certificats et des Arborescences de Listes de révocation

Syntaxe	Nbre de bits	Mnémonique
ECI_certificate_Chain {		
chain_length	8	uimsbf
padding(4)		
for (i=0; i<**chain_length**; i++){		
ECI_RL **rl**		
ECI_certificate **Certificate**		
}		
}		
ECI_RL_Tree {		
ECI_RL **father_revocation_list**		
three_breadth	32	uimsbf
for (i=0; i<**three_breadth**; i++){		
father_node_depth	8	uimsbf
chain_length	8	uimsbf
padding(4)	16	uimsbf
for (i=0; i<**chain_length-1**; i++){		
ECI_certificate **Certificate**		
ECI_RL **rl**		
}		
}		
}		

Sémantique:

chain_length: nombre entier	Longueur de la chaîne.
rl: ECI_RL	**Liste de révocation** du **Certificat** ou du **Père** précédent de la chaîne en cas de première itération d'une chaîne. Les numéros de version du champ d'identificateur des **Listes de révocation** d'une chaîne seront égaux.
Certificate: ECI_certificate	**Père** du **Certificat** suivant dans la séquence en cours.
father_revocation_list: ECI_RL	**Liste de révocation** du **Père** de la chaîne.
three_breadth: nombre entier	Nombre de sous-chaînes dans l'arborescence.
father_node_depth: nombre entier	Niveau du **Certificat de père** dans la **Chaîne de certificats** précédente (comprenant le **Père** de l'arborescence). La liste **Père** héritée est le **Père** de cette chaîne, précédée de son **Père**, etc., jusqu'au **Père** de l'arborescence.

Les règles d'ordre relatives aux **Certificats** dans les arborescences de **Listes de révocation** sont les suivantes:

- Les arborescences ne contiendront pas de **Certificats** en double.
- L'arborescence sera ordonnée de la manière suivante: tous les **Frères** du **Certificat** de la dernière feuille figureront dans la liste sous la forme "chain_length=0 sub-trees" immédiatement après le dernier **Certificat**, puis seront suivis par les sous-arborescences du **Frère** du **Père**, etc.
- Les **Certificats de frères** apparaîtront par ordre d'identificateur dans l'arborescence (le plus bas d'abord).

5.4.2 Règles de traitement relatives aux Chaînes de Certificats

L'**Hôte ECI** vérifie les **Chaînes de Certificats** et se sert du **Système de sécurité évoluée** pour traiter les éléments révoqués de la manière appropriée. Les étapes de sécurité essentielles relatives à la vérification du **Certificat** et de la **Liste de révocation** sont effectuées par le **Système de sécurité évoluée** sécurisé. Celui-ci permet également aux **Clients ECI** de vérifier par la suite la validité des numéros de version de révocation des chaînes qui sont appliqués.

L'**Hôte ECI** peut procéder au traitement itératif des **Chaîne**s **de Certificats**. Le processus commence par la **Liste de révocation** Racine de l'**Autorité de confiance ECI** et finit par le dernier élément d'une chaîne. Le traitement de la **Chaîne de certificats** échoue en cas de défaillance lors des vérifications intermédiaires. En cas d'échec sur une condition, l'**Hôte ECI** fera en sorte que le **Certificat** présent, la **Liste de révocation** et la totalité des **Listes de révocation** et des **Certificats** précédents soient validés par leur signature avant de déclencher les mesures de sa politique relative aux entités révoquées ou aux justificatifs d'identité non valides. Le **Système de sécurité évoluée** défini dans les Recommandations [UIT-T J.1014] et [UIT-T J.1015] veillera à ce que la fiabilité du traitement de la **Chaîne de certificats** soit maintenue.

L'ordre de traitement est indifférent tant qu'il génère le même résultat concernant l'acceptation des chaînes.

1) L'**Hôte ECI** suivra les étapes suivantes de vérification des **Listes de révocation**:
   a) L'**Hôte ECI** vérifiera que le champ **format_version** de la **Liste de révocation** correspond à une version qu'il peut interpréter et veillera à ce que les champs **rl_id.type** et **rl.id_indicator** contiennent les valeurs escomptées.
   b) L'**Hôte ECI** vérifiera si la longueur de la **Liste de révocation** correspond aux valeurs indiquées dans ce champ.
   c) Lorsque le champ **root_version_indicator**=1, l'**Hôte ECI** vérifiera si une Racine est escomptée en tant que **Père** à ce point du traitement de la chaîne, s'il existe une valeur à vérifier pour **root_version** et que la valeur de **min_root_version** ne dépasse pas l'une des versions de Racine utilisées jusqu'à ce point dans le traitement de la chaîne.
   d) L'**Hôte ECI** vérifiera que cette **Liste de révocation** n'a pas été invalidée du fait de son numéro de version minimum par rapport à la **Liste de révocation** précédente de la chaîne ou, en cas de liste de révocation racine, par le numéro min_root_revocation_list utilisé jusqu'à ce point dans le traitement de la chaîne.
   e) L'**Hôte ECI** vérifiera la signature de la **Liste de révocation** avec la clé publique du **Certificat de père**.
   f) L'**Hôte ECI** traitera les éventuels octets d'extension de la **Liste de révocation**, s'il le peut.
   g) L'**Hôte ECI** vérifiera si la **Liste de révocation** a révoqué le <entity type, entity id, version> ***suivant*** de la chaîne et déterminera la version minimum de la liste de révocation à appliquer au **Certificat** concerné.
2) L'**Hôte ECI** suivra les étapes ci-après de prévérification du **Certificat** suivant:
   a) L'**Hôte ECI** vérifiera la version du **Certificat**. Si elle ne correspond pas à ses capacités de traitement, le chargement de la chaîne échouera.
   b) L'**Hôte ECI** vérifiera le champ de type de l'identificateur du certificat et échouera si celui-ci ne correspond pas aux valeurs escomptées.
   c) L'**Hôte ECI** vérifiera que la longueur du **Certificat** correspond à la définition de son format.
   d) L'**Hôte ECI** vérifiera la signature du **Certificat** avec la clé publique du **Certificat de père**.
   e) L'**Hôte ECI** traitera les éventuels octets de champs et/ou d'extension supplémentaires du **Certificat**, s'il le peut.

Une chaîne de **Listes de révocation** extraite d'une arborescence de **Listes de révocation** peut servir à vérifier la révocation d'un élément donné que le **Système de sécurité évoluée** doit charger. Cet élément peut être identifié par la séquence d'identificateurs des **Certificats** utilisée pour le vérifier

lors de son chargement dans le **Système de sécurité évoluée.** Les règles de traitement par défaut des chaînes de **Listes de révocation** seront identiques à celles appliquées aux **Chaînes de Certificats**.

3) Le **Système de traitement des Certificats** chargera la **Liste de révocation** en cours ainsi que les valeurs du champ <entity type, entity id, version> du **Certificat** suivant. Il procédera aux vérifications suivantes:

   a) Le Système de traitement des certificats vérifiera si le champ **format_version** de la **Liste de révocation** correspond à une version qu'il peut interpréter et si les champs **rl_id.type** et **rl.id_indicator** contiennent les valeurs escomptées.

   b) Si le **Père** est un **Certificat racine** (**root_version_indicator**=1), le Système de traitement des certificats sélectionnera comme **Père** le **Certificat racine** avec **root_version**. Sinon, il utilisera le **Certificat** préchargé ou précédent.

   c) Le Système de traitement des certificats vérifiera la signature de la **Liste de révocation** avec la clé publique du **Certificat de père**.

   d) Le Système de traitement des certificats vérifiera si la longueur de la **Liste de révocation** correspond aux valeurs indiquées dans ce champ.

   e) Le Système de traitement des certificats vérifiera que le numéro de version de la **Liste de révocation** n'a pas été invalidé.

   f) Le Système de traitement des certificats vérifiera si la **Liste de révocation** n'a pas révoqué le champ <entity type, entity id, version> ***suivant*** dans la chaîne et déterminera la version minimum de la liste de révocation à appliquer au **Certificat** concerné.

4) Ensuite, l'**Hôte ECI** chargera le **Certificat** dans l'emplacement approprié du Système de traitement des certificats, qui effectuera les vérifications suivantes:

   a) Le Système de traitement des certificats vérifiera si le champ **format_version** de la **Liste de révocation** correspond à une version qu'il peut interpréter et si les champs **id.type** et **id.entity_id** contiennent les valeurs escomptées.

   b) Le Système de traitement des certificats vérifiera si la longueur du **Certificat** correspond aux valeurs indiquées dans ce champ.

   c) Le Système de traitement des certificats vérifiera la signature en la comparant à la clé publique du **Certificat de Père**.

5.5 Ensembles d'arborescences de révocation et fichiers de données de révocation

Les données de révocation utilisées pour vérifier une **Entité** donnée impliquent la sélection des données de révocation contenues dans la liste de révocation du **Père** de l'**Entité** en question.

Lors de la distribution des données de révocation, il est possible de constituer une arborescence avec les chaînes utilisées pour révoquer plusieurs entités cibles. Ce processus évite de dupliquer les **Certificats d'Enfants** et Racine ainsi que les **Listes de révocation** associées et permet de procéder à des recherches plus ordonnées dans les **Équipements CPE**.

Afin de faciliter l'assemblage et le désassemblage des données de révocation, il suffit de combiner les arborescences de révocation pour constituer un ensemble d'arborescences. Cependant, à l'exception de la liste de révocation commune du **Père**, les ensembles d'arborescences ne se recouperont pas. Les ensembles d'arborescences peuvent contenir plusieurs listes de révocation de Racine (pendant le déploiement d'un changement de Racine d'**Autorité de confiance**).

La définition des **Chaînes de Certificats** et des Arborescences de **Listes de révocation** sera conforme au Tableau 5.5-1.

Tableau 5.5-1 – Définition des ensembles d'arborescences de Listes de révocation

Syntaxe	Nbre de bits	Mnémonique
ECI_RL_Tree_Set {		
tree_number	32	uimsbf
for (i=0; i<tree_number; i++) {		
ECI_RL_Tree **tree**		
}	8	uimsbf
}		

Sémantique:

tree_number: nombre enter	Nombre d'arborescences composant l'ensemble.
tree: ECI_RL_Tree	Arborescence (dont **Certificat racine**) des **Certificats** et de leurs **Listes de révocation**.

NOTE – Les serveurs en ligne peuvent distribuer des arborescences ciblant une seule **Entité** (de fait, des chaînes) pour réduire le trafic des données. Sur les réseaux de radiodiffusion, il est possible de fractionner et de fusionner aisément les arborescences afin qu'elles correspondent au nombre de seaux (voir le § 7.7.2) utilisés dans le carrousel de transmission.

Les arborescences ou les ensembles d'arborescences de révocation ne doivent pas nécessairement contenir toutes les entités d'une classe. L'**Opération de plate-forme** peut composer l'ensemble d'arborescences de révocation à sa guise en veillant à limiter au maximum le risque dans les **Équipements CPE** déployés dans son réseau. Dans les réseaux de radiodiffusion, on peut également alterner les **Listes de révocation** en temps voulu afin d'étendre la couverture de la révocation.

L'interface **ECI** requiert que les **Équipements CPE** stockent en permanence des chaînes d'**autorités de confiance ECI** pour tous les éléments susceptibles d'être téléchargés afin que les entités révoquées le demeurent. Cette obligation est spécifiée dans les paragraphes y afférents.

Pour la commodité du transport, les ensembles d'arborescences de révocation **ECI** sont regroupés dans le format indiqué dans le Tableau 5.5-2.

Tableau 5.5-2 – Fichier de données de révocation

Syntaxe	Nbre de bits	Mnémonique
ECI_revocation_data_file {		
magic = 'ERD'	24	uimsbf
version:	8	uimsbf
father_type	4	uimsbf
sub_type	4	uimsbf
ECI_RL_Tree_Set **revocation_data**		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations ASCII 8 bits des caractères 'ERD'. L'**Hôte ECI** contrôlera la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu afin de renforcer l'intégrité des données.
version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. L'**Hôte ECI** ignorera les images dont il ne reconnaît pas le numéro de version.
father_type: nombre entier	Type du **Père** commun aux données de la **Liste de révocation**. 0x0 correspond au **Certificat racine** de l'interface **ECI**. Les valeurs 0x1-0x7 sont réservées. Les valeurs 0x8-0xF pourront être utilisées pour des applications privées.
sub_type: nombre entier	Si le champ father_type est égal à 0x0, il définit le type de **Liste de révocation** commune conformément au Tableau 5.2-2 du **Certificat racine ECI** des données de révocation. Cette valeur n'est pas définie pour les autres valeurs du champ father_type.
revocation_data: ECI_RL_Tree_Set	Ensemble d'arborescences de listes de révocation pour les éléments révoqués.

5.6 Signatures des éléments de données de grande taille

L'interface **ECI** calcule les signatures des éléments de données de grande taille (par exemple, images logicielles) à l'aide d'une fonction de hachage des données en masse conjointement à une opération de signature normale. Le Tableau 5.6-1 définit la signature des éléments de données de grande taille.

Tableau 5.6-1 – Définition de la signature des éléments de données de grande taille

Syntaxe	Nbre de bits	Mnémonique
`ECI_Data_Signature {`		
`    sign_version`	8	uimsbf
`    padding(4)`	24	uimsbf
`    if (sign_version == 0x01){`		
`        for (i=0; i<256; i++){`		
`            signature_byte`	8	uimsbf
`        }`		
`    }`		
`}`		

Sémantique:

sign_version: nombre entier	Version de signature. La valeur 0x01 est la version présente. Toutes les autres valeurs de version sont réservées. Les **CPE** qui n'ont pas mis en œuvre de version ignoreront ce champ (et les éventuelles données suivantes).
signature_byte: octet	Séquence d'octets représentant la signature d'un élément de grande taille.

L'algorithme de signature est défini dans l'Annexe A.

5.7 Certificats racine

5.7.1 Définition d'un Certificat racine

L'interface **ECI** utilise une séquence de ***versions*** de **Certificat racine**. L'**Autorité de confiance ECI** peut commencer par utiliser une nouvelle version du **Certificat racine**, par exemple si l'une des **Listes de révocation** antérieures de l'un des **Enfants** est trop volumineuse ou si la clé secrète associée à la clé publique du **Certificat** n'est plus considérée comme suffisamment secrète.

Le **Certificat racine** utilise le champ d'identificateur des **Certificats ECI** avec la définition donnée dans le Tableau 5.7-1. Les champs de type et d'identificateur ne sont jamais utilisés. Seul le champ de version est à prendre en compte.

Tableau 5.7-1 – Définition du champ ECI Root_id

Syntaxe	Nbre de bits	Mnémonique
ECI_Root_Id {		
type /* voir le Tableau 5.2-1***/**	4	uimsbf
id /* voir le Tableau 5.2-2 ***/**	20	uimsbf
version:	8	uimsbf
}		

Sémantique:

version: nombre entier	Numéro de version du **Certificat**. La numérotation commence à 1 et augmente de 1 à chaque nouvelle émission d'un **Certificat racine**. La valeur 0x00 est réservée.

5.7.2 Gestion des Certificats racine de l'Hôte ECI

L'**Autorité de confiance ECI** peut commencer par utiliser un nouveau **Certificat racine** doté d'un numéro de version supérieur. A un moment ultérieur quelconque, elle peut émettre une **Liste de révocation** pour le nouveau **Certificat racine** afin de révoquer les **Certificats racine** antérieurs. Cette action invalide tous les **Certificats** signés par la Racine concernée.

Alternativement, l'**Autorité de confiance ECI** peut décider que la **Liste de révocation** d'un type d'entités particulier (par exemple, **Fabricants**) est de trop grande taille et décider de réémettre de nouvelles versions de tous les **Certificats** antérieurs en utilisant un champ **min_id_version** de valeur supérieure dans la **Liste de révocation** du type d'**Entité** concerné. Cette action invalide de façon effective tous les **Certificats** émis pour le type d'**Entité** concerné jusqu'à **min_entity_version**-1. En général, afin de remplacer les **Certificats** révoqués, elle nécessite d'émettre un grand nombre de nouveaux **Certificats** dotés d'un numéro de version supérieur pour les entités qui continuent à utiliser une version de **Certificat** inférieure.

Des ressources de stockage des **Certificats racine** que l'**Hôte ECI** fournira sont proposées dans le document [b-UIT-T J Suppl. 7].

6 Chargeur d'hôte ECI

6.1 Introduction

Le processus de chargement d'un **Hôte ECI** comporte les aspects suivants:

1) Stockage d'une image, vérification de son authenticité par l'**Équipement CPE** à l'aide des données d'authentification fournies par l'**Autorité de confiance ECI** puis activation de l'image.
2) Format du ou des fichiers contenant l'image et toutes les autres informations requises pour charger l'image dans l'**Équipement CPE**.
3) Protocole de transport pour livrer l'**Image de l'Hôte ECI** à l'**Équipement CPE**, comprenant l'éventuelle découverte par ce dernier de l'emplacement des images requises ainsi que le stockage éventuel des images transportées, de la chaîne de validation **ECI** complémentaire et des données de signature.
4) Toute révocation éventuelle des **Images de l'hôte ECI** propre à l'**Opérateur**. Le format de ces données est défini dans le § 6, et leur application dans le § 8.

La logique de vérification et d'authentification des images s'appliquera aux **Images de l'hôte ECI** récemment téléchargées et aux données d'authentification à chaque redémarrage de l'**Équipement CPE** ainsi que pendant le fonctionnement normal de l'**Équipement CPE**, lorsque cela est prévu.

6.2 Stockage, vérification et activation

6.2.1 Principes de fonctionnement

L'**Hôte ECI** fait en sorte que les **Clients ECI** puissent s'exécuter dans un environnement privé et sans risque d'altérations, conformément aux exigences en matière de fiabilité de l'interface **ECI** concernant la mise en œuvre de ce type de clients. L'**Hôte ECI** empêche également les interférences entre **Clients ECI**. Dans ce but, l'**Autorité de confiance ECI** pourra certifier les logiciels des **Équipements CPE** et le chargeur de ces équipements vérifiera l'authenticité des images logicielles qu'il charge.

De nombreux **Équipements CPE** utilisent des chargeurs à plusieurs étapes. L'interface **ECI** part du principe que la puce principale de l'**Équipement CPE** charge plusieurs images d'initialisation propres à la puce avant de commencer à charger les images logicielles normales. Ce type d'images peut être implicitement certifié dans le cadre de l'accord de licence du fournisseur de puces octroyé par l'**Autorité de confiance ECI**. Ou bien elles peuvent être incluses dans le processus de certification du **Fabricant** défini dans le présent paragraphe.

Si le logiciel de l'une des images gérées par l'interface **ECI** présente un problème de sécurité à un stade ultérieur, l'**Autorité de confiance ECI** et le **Fabricant de l'Équipement CPE** pourront le révoquer et le remplacer par une version exempte de tout problème.

La Figure 6.2.1-1 suppose que **Img1** est une image propre à une puce requise pour que celle-ci puisse commencer à charger des images d'application normales. Cette image est protégée par une signature propre à la puce, **CS1**, vérifiée par le **chargeur de puces** à l'aide d'une clé propriétaire du fournisseur de puces.

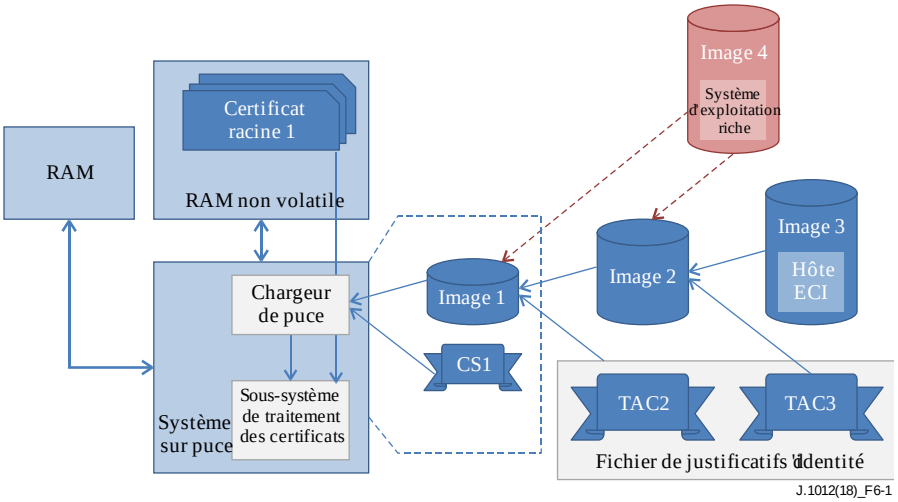


Figure 6.2.1-1 – Exemple de processus de chargement d'un Hôte ECI

Quand **Img1** est en cours d'exécution, la puce procède au chargement d'autres images. Elle charge **Img2** qui peut être authentifiée par une **Chaîne de certificats** et la signature d'image **TAC2**. L'image est vérifiée par le **Certificat racine** de l'**Autorité de confiance**, le **Sous-système de traitement des Certificats** et la signature **TAC2. Img2** charge **Img3** qui contient le logiciel de l'**Hôte ECI**. **Img3** est vérifiée par le **Sous-système de traitement des Certificats**, les **Certificats racine,** la **Chaîne de certificats** de l'autorité de confiance et la signature d'image **TAC3**. Les autres images telles **Img4** contenant, par exemple, un système d'exploitation riche, non certifiées par l'**Autorité de confiance ECI** pourront être chargées si cela ne remet pas en cause la sécurité de l'**Hôte ECI**.

Les justificatifs d'identité attribués aux images par l'autorité de confiance sont transportés dans un fichier spécial de justificatifs d'identité.

L'**Autorité de confiance ECI** certifie l'intégrité de la sécurité de l'**Hôte ECI**: sa capacité à assurer la confidentialité des clients, à résister aux tentatives d'altération par des menaces extérieures à l'**Hôte**

ECI et à empêcher les interférences indésirables entre les clients. Afin de renforcer la protection du chargement des images, les **Fabricants d'Équipements CPE** souhaiteront éventuellement utiliser des mesures de sécurité complémentaires faisant appel à leurs propres procédures de chiffrement et d'authentification.

Les opérations de plate-forme peuvent vérifier l'actualité des **Images de l'hôte ECI** et décider de ne pas déchiffrer les services. Dans ce but, le Système de traitement des certificats extrait le numéro de version minimum des **Listes de révocation** utilisées pour vérifier les éléments chargés, ce qui permet aux **opérations de plate-forme** de vérifier si une **Liste de révocation** récente a été appliquée. Les procédures d'acceptation propres aux **opérations de plate-forme** pour un **Hôte ECI** sont définies au § 8.

Le **Chargeur d'hôte ECI** stockera les **Images de l'hôte ECI** les plus récentes et leurs justificatifs d'identité les plus récents dans la mémoire RAM non volatile. Le **Chargeur d'hôte ECI** vérifiera à nouveau chaque image qu'il charge lors du redémarrage de l'**Hôte ECI.** Cette procédure confirme l'authenticité de l'**Hôte ECI** à chaque redémarrage.

6.2.2 Définition des justificatifs d'identité

6.2.2.1 Certificats relatifs aux Images de l'hôte ECI

Concernant la diversité des **Images de l'hôte ECI,** l'interface **ECI** distingue deux types d'**Équipements CPE** conformes **ECI**:

1) les **Équipements CPE** génériques qui chargeront le même jeu d'**Images de l'hôte ECI** sur chaque instance du même type et de la même version d'**Équipement CPE**;
2) les **Équipements CPE** individualisés qui chargeront un ensemble (partiellement) différent d'images sur chaque **Équipement CPE** du même type et de la même version. Une série d'images de même "type" mais individualisée sur chaque **Équipement CPE** s'appelle une **Série d'images**.

La **Chaîne de certificats de l'Hôte ECI** est composée des **Certificats** suivants (chacun certifié par son prédécesseur):

1) certificat racine:
   - Il représente l'**Entité** racine de l'**Autorité de confiance ECI**. Sa clé publique servira à la vérification.
2) certificat de Fabricant:
   - Il représente l'**Entité Autorité de confiance ECI** d'un **Fabricant** donné. Sa clé publique servira à la vérification.
3) certificat d'Hôte:
   - Il représente un **Équipement CPE** physique et une version logicielle d'**Hôte ECI** certifiés par l'**Autorité de confiance ECI.** La clé publique de ce **Certificat** servira à authentifier toutes les **Images de l'hôte ECI** des **Hôtes ECI** génériques. Elle servira également à vérifier les **Images de l'hôte ECI** "individualisées".
4) certificat de Série d'images d'hôte
   - Cette **Entité** approuve de façon générique une série d'images propres à une configuration particulière d'**Équipement CPE**, mais identiques du point de vue de l'**Autorité de confiance ECI**. Dans le cas des **Hôtes ECI** individualisés, la clé publique de ce **Certificat** servira à authentifier l'**Image de l'Hôte ECI** destinée à un **Équipement CPE** particulier dont l'identificateur correspond à celui figurant dans le **Certificat**.

NOTE – Chaque identificateur d'Entité doit être interprété dans le contexte de l'Entité autorisatrice. En d'autres termes, les identificateurs sont relatifs.

L'**Image de l'Hôte ECI** et la structure de certification associée sont décrites dans la Figure 6.2.2.1-1 et le Tableau 6.2.2.1-1 donne une vue d'ensemble des paramètres concernés.

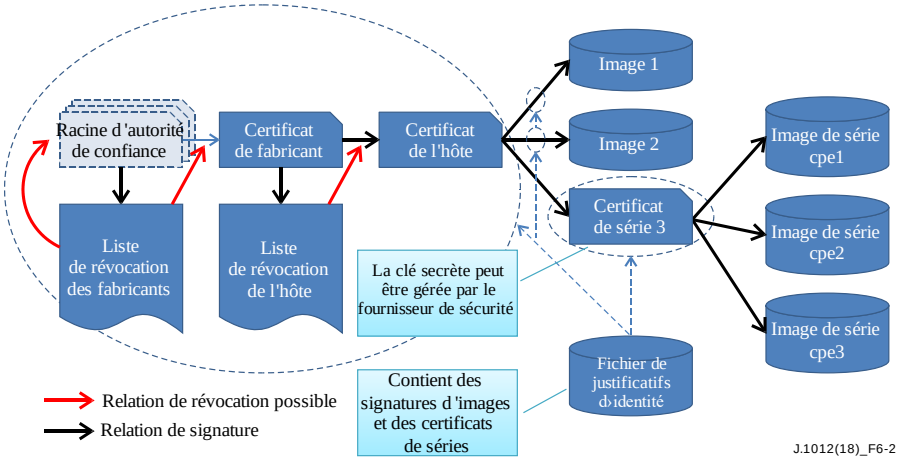


Figure 6.2.2.1-1 – Structure de Certification des images d'Hôte ECI

Tableau 6.2.2.1-1 – Vue d'ensemble des paramètres des Certificats associés aux Hôtes ECI

Type	Entité	Valeur du champ d'identificateur du Certificat	Traitement particulier par l'Hôte ECI
0x0	**Fabricant**	manufacturer_id, version	L'identificateur du Fabricant sera comparé à celui figurant dans le Bloc de sécurité évoluée de l'**Équipement CPE**.
0x0	Hôte	cpe_type, cpe_model, host_version	Le Bloc de sécurité évoluée comparera les champs cpe_type et cope_model au type et au modèle de l'**Équipement CPE**.
0x8	**Série d'images** d'équipement CPE	target_id	Le champ target_id sera comparé à l'identité de l'**Équipement CPE**.
0x8	Image d'**équipement** CPE	s. o.	
0x8	**Image de l'Hôte ECI**	ECI_Host_Image_Id	Il s'agit du type de la signature réelle de l'image.

Les définitions des **Certificats** associés à l'**Hôte ECI** seront conformes au **Certificat ECI** (ECI_certificate) général défini au § 5.2. La définition des champs d'identificateur des **Certificats** destinés à la gestion des **Hôtes ECI** est fournie dans le Tableau 6.2.2.1-2.

Tableau 6.2.2.1-2 – Définition des champs d'identificateurs des Certificats associés aux Hôtes

Syntaxe	Nbre de bits	Mnémonique
ECI_Manufacturer_Id {		
padding(4)		
type /* voir le Tableau 5.2-2 */	4	uimsbf
manufuacturer_id	20	uimsbf
Version	8	uimsbf
}		
ECI_CPE_Type_ID {		
cpe_type	12	uimsbf
cpe_model	8	uimsbf
}		
ECI_Host_Id {		
padding(4)		
type /* voir le Tableau 5.2-2 */	4	uimsbf
ECI_CPE_Type_Id **cpe_type_id**	20	uimsbf
host_version	8	uimsbf
}		
ECI_Host_Image_Series_Id {		
padding(4)		
type /* voir le Tableau 5.2-2 */	4	uimsbf
image_series_model	8	uimsbf
image_series_model_extension	4	uimsbf
image_series_version	16	uimsbf
}		

Sémantique:

type	Valeur conforme au Tableau 5.2-2.
manufacturer_id: nombre entier	Identificateur attribué au **Fabricant** par l'**Autorité de confiance ECI**
cpe_type: nombre entier	Identificateur attribué au modèle d'**Équipement CPE** par l'**Autorité de confiance ECI**. Les valeurs 0x000 et 0x3F0..0x3FF sont réservées. Les **Équipements CPE** du même modèle auront de nombreux points communs et utiliseront la même technologie de sécurité **ECI**.
cpe_model: nombre entier	Identificateur attribué à une version d'un modèle donné identique à de nombreux titres mais présentant plusieurs différences significatives. Sa valeur est attribuée par l'**Autorité de confiance ECI**. Les valeurs 0x00 et 0xF0..0xFF sont réservées.
cpe_type_id: ECI_CPE_Type_id	Identificateur du type d'**Équipement CPE** (version + modèle); unique dans le contexte du champ manufacturer_id.
cpe_host_version	Identificateur attribué à un ensemble d'images constituant une configuration d'**Hôte ECI** pour l'**Équipement CPE.**
image_series_model: nombre entier	Identificateur des images de même type pour les **Équipements CPE** prenant en charge les **Séries d'images**, la distinction étant faite par le champ **cpe_id**. Les valeurs 0x000 et 0xF00...0xFFF sont réservées.
image_series_version: nombre entier	Identificateur attribué de façon incrémentielle à la version du modèle de **Série d'images** par l'**Autorité de confiance ECI**. Les valeurs 0x0000 et 0xF000...0xFFFF sont réservées.

6.2.2.2 Signatures des images d'Hôte ECI

L'identificateur des **Images de l'hôte ECI** sera égal à celui des Séries d'images d'hôte. Il est défini dans le Tableau 6.2.2.2-1.

Tableau 6.2.2.2-1 – Définition des identificateurs d'images d'hôte et de Séries d'images d'hôte

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_Image_Id {		
padding(4)		
type /* voir le Tableau 5.2-2 */	4	uimsbf
image_model	8	uimsbf
image model_extension	4	uimsbf
image_version	16	uimsbf
}		
ECI_CPE_Id {		
cpe_serial_number	28	uimsbf
cpe_type	12	uimsbf
manufacturer_id	20	uimsbf
}		
ECI_Image_Target_Id {		
padding(4)		
target type	4	uimsbf
if (target type == 0x1){		
ECI_CPE_Id **cpe_id**	60	uimsbf
}		
}		

Sémantique:

type	Valeur conforme au Tableau 5.2-2.
image_model: nombre entier	Identificateur attribué à une **Image de l'Hôte ECI** ou à une série d'images se remplaçant mutuellement. Les valeurs 0x00 et 0xF0..0xFF sont réservées.
image_model_extension: nombre entier	Extension du champ ci-dessus. Dans les applications normales, ce champ doit être mis à 0x0.
image_version: nombre entier	Version d'une image de même type attribuée de façon incrémentielle. Les valeurs 0x00 et 0xF0..0xFF sont réservées.
cpe_serial_number: nombre entier	Numéro de série de l'**Équipement CPE** auquel l'image est destinée. Le champ cpe_serial_number sera unique dans le contexte du champ <manufacturer_id, cpe_type_id>.
cpe_type: nombre entier	Champ cpe_type défini dans la structure ECI_CPE_Type_Id.
manufacturer_id: nombre entier	Champ manufacturer_id défini dans la `structure` ECI_Manufacturer_Id.
target type: nombre entier	Type d'identification de cible pour l'image de la série. La valeur 0x1 définit cette structure et indique qu'un identificateur cpe_id est utilisé comme cible. Les autres valeurs sont réservées.
cpe-id: ECI_CPE_Id	Identificateur de l'**Équipement CPE** cible d'une image (d'**Hôte ECI** ou de **Client ECI**) dans une série.

Les signatures d'**images d'hôtes ECI** et de **Séries d'image d'Hôte ECI** utilisées pour signer les **Images de l'hôte ECI** réelles se serviront de la structure de signature des données de grande taille définie au § 5.5.

6.2.2.3 Justificatifs d'identité de l'Hôte ECI

Le Tableau 6.2.2.3-1 définit la structure des justificatifs d'identité de l'**Hôte ECI** qui vérifie un ensemble d'**Images de l'hôte ECI**.

Tableau 6.2.2.3-1 – Définition de la structure des justificatifs d'identité de l'Hôte ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_Credentials{		
image_credential_version	8	uimsbf
if (image_credential_version == 0x01) {		
padding(4)	24	uimsbf
ECI_certificate_Chain **image_chain**		
nr_images	8	uimsbf
padding(4)	24	uimsbf
for (i=0; i<images; i++){		
ECI_Host_Image_Id **image_id**	32	uimsbf
if (image_id.type == 0x8) {		
ECI_certificate **series_cert**		
} else if (image_id.type == 0x9){		
ECI_Data_signature **image_signature**		
}		
}		
Extension_Field **extension**		
}		
}		

Sémantique:

image_credential_version: octet	Version du format des justificatifs d'identité. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. Les **chargeurs d'Hôte ECI** ignoreront les justificatifs d'identité dont ils ne reconnaissent pas les valeurs.
image_chain: ECI_certificate_Chain	**Chaîne de certificats** à deux niveaux de profondeur commençant par la **Liste de révocation de Racine des Fabricants** et allant jusqu'au **Certificat d'Hôte ECI**. Le dernier **Certificat** servira à vérifier la signature des images de tout **Certificat de série d'images**.
nr_images: nombre entier	Nombre d'images dont les signatures sont incluses.
image_id	Identificateur de l'image dont la signature suit dans la boucle. Les identificateurs d'images figurant dans la boucle auront des valeurs différentes pour le champ **image_id.image_model.**
series_cert: ECI_certificate	Certificat servant à vérifier une **Série d'images**.
image_signature: ECI_Data_Signature	Signature de l'image (y compris identificateur de l'Image de l'Hôte).
extension: champ d'extension	Champ d'extension compatible avec les versions antérieures.

Lors de la vérification du champ **image_chain**, l'**Équipement CPE** respectera les règles de traitement génériques applicables aux chaînes définies dans le § 5.4.

6.2.3 Processus de chargement d'un fichier d'Image de l'Hôte ECI

L'**Équipement CPE** stockera, vérifiera et activera l'exécution de l'ensemble de fichiers d'**Images de l'hôte ECI** requis pour démarrer l'**Hôte ECI**. En général, l'**Image de l'hôte ECI** est activée lors du démarrage de l'**Équipement CPE**.

L'**Équipement CPE** utilisera une fonction de traitement fiable appelée **Chargeur d'hôte ECI** pour télécharger, vérifier et activer l'**Image de l'Hôte ECI** choisi. Si, par exemple, l'image de démarrage de l'**Équipement CPE** contenant le **Chargeur d'hôte ECI** commence l'exécution d'une deuxième image et que celle-ci charge et démarre l'exécution d'une troisième image, la fonctionnalité permettant à la deuxième image de charger convenablement la troisième vérifiant la signature de l'image sera considérée comme une fonctionnalité du **Chargeur d'hôte ECI** pour l'**Équipement CPE** en question. Seule la fonction de **Chargeur d'hôte ECI** peut vérifier et démarrer une **Image d'hôte ECI**. Le **Chargeur d'hôte ECI** utilisera le **Système de traitement des certificats** pour vérifier les justificatifs d'identité de l'image.

L'**Équipement CPE** stockera l'ensemble le plus récent de fichiers d'**images d'Hôte ECI** et ses justificatifs d'identité, qu'il a téléchargés dans la **Mémoire non volatile**. Lors du démarrage de l'**Équipement CPE**, le **Chargeur d'hôte ECI** sera capable de localiser les images et de commencer à les charger d'une manière adaptée au type d'**Équipement CPE** concerné.

À l'aide du Système de traitement des certificats, le **Chargeur d'hôte ECI** utilisera les règles normales de traitement des chaînes énoncées au § 5.4 pour vérifier chaque image chargée. Les images génériques et les **Certificats de séries d'images** seront vérifiés à l'aide de la clé publique du **Certificat** de l'Hôte. La clé publique du **Certificat de série d'images** servira à vérifier les **images de la série** et l'**Équipement CPE** comparera son champ cpe_id à celui de l'image.

Si l'image est compromise (échec de la vérification de la signature par le Système de traitement des certificats), le **Chargeur d'hôte ECI** la refusera et l'**Équipement CPE** ne pourra pas créer d'instance de l'**Hôte ECI** sur le matériel concerné. L'**Équipement CPE** sera capable de se rétablir: il sera doté d'une procédure de rétablissement permettant de réinitialiser l'**Image de l'Hôte ECI** la plus récente et ses justificatifs d'identité, par exemple, en rechargeant le dernier ensemble en date de fichiers d'**Images de l'hôte ECI** à partir du canal de radiodiffusion, depuis le serveur d'**Images de l'hôte ECI** en ligne ou par un autre moyen.

L'**Hôte ECI** stockera les dernières versions des **Certificats** de chaînes d'**Hôtes ECI** qu'il a acquises, quel que soit le canal utilisé. Cette fonctionnalité "verrouille" le dernier **Certificat** d'Hôte disponible en tant que base des futures vérifications d'images.

La séquence de chargement des **Images de l'hôte ECI** n'est pas directement vérifiée par le processus de vérification des signatures: celui-ci sera exécuté par le chargeur d'amorce pour la première **Image de l'Hôte ECI** et, pour les activations suivantes, par les **Images de l'hôte ECI** précédentes elles-mêmes.

6.3 Formats de fichiers associés aux Hôtes ECI

La présente Recommandation ne définit pas l'attribution des noms de fichiers ou les autres attributs des métadonnées des fichiers d'**Images de l'hôte ECI**. Il gère les données d'**Images de l'hôte ECI** sous la forme d'un ensemble de conteneurs de données (fichiers sans nom compatibles avec l'interface ECI) identifiés par leur identificateur d'Image d'hôte et les justificatifs d'identité de l'interface ECI (**Chaînes de Certificats** et signatures) requis pour les authentifier.

Les fichiers d'**Images de l'hôte ECI** constitueront une séquence ECI_host_image_header et contenu de l'image. Ils suivront la définition donnée dans le Tableau 6.3-1.

Tableau 6.3-1 – Définition des fichiers d'Images de l'hôte ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_Image_File {		
magic = 'EHI'	24	
image_header_version	8	uimsbf
if (image_header_version == 0x01) {		
ECI_Host_Image_Id **host_image_id**	32	uimsbf
ECI_Manufacturer_Id **manufacturer_id**	32	uimsbf
Extension_Field **extensions**		
for (i=0; i<n; i++) {		
host_image_byte	8	uimsbf
}		
}		
}		

Sémantique:

host_image_byte: octet	Format réel d'**Image de l'Hôte ECI** propre à l'**Équipement CPE**.
magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères `'EHI'`. Le micrologiciel de l'**Équipement CPE** contrôlera la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu et renforcer l'intégrité des données.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées.
host_image_id: ECI_Host_Image_Id	Identificateur d'**Image de l'Hôte ECI** de l'image. Les **Équipements CPE** vérifieront ce champ avant de charger une (nouvelle) **Image de l'Hôte ECI**.
manufacturer_id ECI_Manufacturer_Id	**ECI**´_Manufacturer_ID du **Fabricant** de l'**Équipement CPE** de l'**Image de l'Hôte ECI**. Les **Équipements CPE** vérifieront ce champ avant de charger une (nouvelle) **Image de l'Hôte ECI**. Voir la NOTE.
extensions: Extension_Field	Voir le § 5.1 de la présente Recommandation: extensions compatibles avec les versions antérieures.
host_image_byte: octet	**Image de l'Hôte ECI.**
NOTE – Ce champ doit également correspondre à l'identificateur unique d'organisation (OUI) du **Fabricant** utilisé dans les carrousels de radiodiffusion pour transporter le fichier associé.	

Les fichiers de **Séries d'images** possèdent une signature unique transportée dans le fichier d'images lui-même. Par conséquent, un format de fichier donné respectera la définition donnée dans le Tableau 6.3-2.

Tableau 6.3-2 – Définition des fichiers de Séries d'Images de l'hôte ECI

Syntaxe	Nbre de bits	Mnémonique
`ECI_Host_Image_Series_File {`		
`  magic = 'EHS'`		
`  image_header_version`	8	uimsbf
`  if (image_header_version == 0x01) {`		
`  ECI_Data_Signature image_signature`		
`  ECI_Image_Target_Id target_id`	64	
`  Extension_Field extensions`		
`  for (i=0; i<n; i++) {`		
`  host_image_byte`	8	uimsbf
`  }`		
`  }`		
`}`		

Sémantique:

host_image_byte: octet	Format réel d'**Image de l'Hôte ECI** propre à l'**Équipement CPE**.
magic: octet[10]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères `'EHS'`. Le micrologiciel de l'**Équipement CPE** contrôlera la valeur de ce champ pour vérifier si un fichier ECI possède le format attendu et renforcer l'intégrité des données.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées.
image_signature: ECI_Data_Signature	Signature sur toutes les données suivantes du fichier d'images.
target_id: ECI_Series_Image_Target_Id	Identificateur cible de l'image. La valeur du champ target_id.target_type est 0x01. Toutes les autres valeurs sont réservées.
extensions: Extension_Field	Voir le § 5.1: extensions compatibles avec les versions antérieures.
host_image_byte: octet	Séquence d'octets constituant l'**Image** de l'Hôte.

Les justificatifs d'identité de l'**Image de l'Hôte ECI** suivent la définition du Tableau 6.3-3, qui, pour l'essentiel, est la **Chaîne de certificats** comportant l'ensemble de signatures d'images ou les **Certificats de séries d'images**.

Tableau 6.3-3 – Définition des fichiers de justificatifs d'identité des images d'Hôte ECI

Syntaxe	Nbre de bits	Mnémonique
`ECI_Host_Image_Credential_File{`		
`  magic = 'EHC'`	24	uimsbf
`  version:`	8	uimsbf
`  if (version == 0x01) {`		
`  ECI_Host_Credentials credentials`		
`  }`		
`}`		

Sémantique:

magic	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères `'EHC'`. Le micrologiciel de l'**Équipement CPE** contrôlera la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu et renforcer l'intégrité des données.
version:	Version du format du fichier. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées.
credentials: ECI_Host_Credentials	Justificatifs d'identité d'une **Image de l'Hôte ECI** ou d'un groupe d'**Images de l'hôte ECI**.

L'identificateur host_image_id sert à identifier les signatures de l'**Autorité de confiance de l'interface ECI** pour un ensemble de fichiers d'**Images de l'hôte ECI** comprenant un téléchargement complet dans la structure de justificatifs d'identité de l'interface **ECI**.

Les **Équipements CPE** conformes **ECI** sont autorisés à télécharger d'autres modules logiciels d'**Équipement CPE** propriétaires avec le même protocole de transport que celui utilisé pour les fichiers d'**Images de l'hôte ECI.** Aucun format particulier n'est requis pour ces images.

Sur les supports de radiodiffusion, il est commode de distribuer les données de révocation de nombreux **Hôtes ECI** sous la forme d'un unique fichier de grande taille. Les **Hôtes ECI** recevant ces données peuvent s'en servir pour vérifier leur propre **Certificat d'Hôte ECI.**

Le fichier de données de révocation de l'**Hôte ECI** utilise le format ECI_Revocation_Data_File défini dans le Tableau 5.5-2. Le fichier de données de révocation de l'**Hôte ECI** utilise un champ father_type égal à 0x0 (**Certificat racine**) et un champ sub_type égal au type de liste de révocation du **Fabricant**.

Le champ revocation_data respecte la contrainte imposant que la liste de révocation feuille des arborescences corresponde à des listes de révocation d'**Hôtes ECI**.

6.4 Protocoles de transport d'Images de l'hôte ECI

6.4.1 Introduction

La présente Recommandation distingue trois types de fourniture des **images** d'Hôtes:

1) **Radiodiffusion**: l'interface **ECI** définit les protocoles autorisant les **Opérateurs de plate-forme** à signaler et à fournir de nouveaux fichiers d'**Hôtes ECI** du **Fabricant de l'équipement CPE** aux **Équipements CPE** dans le champ utilisant DVB-SSU.

2) **En ligne**: l'interface **ECI** autorise les **Équipements CPE** connectés à Internet à télécharger des fichiers d'**Images de l'hôte ECI** avec n'importe quel protocole propriétaire, mais suggère d'utiliser HTTP 1.1 et une interface définie **ECI** vers un serveur web fourni par un opérateur.

3) **Autre**: les **Fabricants d'équipements CPE** et/ou les **Opérateurs** peuvent également recourir à d'autres modes de fourniture de fichiers **d'Images de l'hôte ECI**, y compris hors ligne par le biais d'une clé USB, par exemple. Ce moyen de transport des images ne relève pas du champ d'application de la présente Recommandation. Néanmoins, les images chargées avec ce protocole seront conformes à la vérification du format des fichiers et des images exposées dans les § 6.2 et 6.3.

Les **Équipements CPE** conçus pour acquérir des **Services** à partir de réseaux de radiodiffusion numérique mettront en œuvre le protocole de transport de la radiodiffusion pour les **Images de l'hôte ECI** tel que défini dans le § 6.4.2**.**

Les **Équipements CPE** dotés d'une connexion IP mettront en œuvre un protocole de transport Internet pour les **Images de l'hôte ECI** en ligne tel que défini dans le § 6.4.3 ainsi que le protocole défini dans le § 7.7.3.3.

Un **Équipement CPE** pourra mettre en œuvre n'importe quel protocole de transport d'**Images de l'hôte ECI**, y compris les protocoles de radiodiffusion et de transport hors connexion des **Hôtes ECI** (clés USB, par exemple). Dans tous les cas, le **Fabricant d'équipement CPE** veillera à mettre en place les moyens pratiques permettant de mettre à jour l'**Hôte ECI** sur le terrain grâce à la combinaison des protocoles de transport ci-dessus, en tenant compte des cas d'utilisation pratiques où certaines connexions du réseau ne sont pas établies.

6.4.2 Protocole de transport des Hôtes ECI via la radiodiffusion

6.4.2.1 Généralités et profil

Le protocole de transport des **Hôtes ECI** via la radiodiffusion permet aux nouveaux fichiers d'**Images de l'hôte ECI** et aux données associées d'être transportés depuis un **Fabricant d'équipement CPE** jusqu'à un **Équipement CPE** via une infrastructure tête de réseau de radiodiffusion de l'**Opérateur**. Le protocole permet également de transporter les fichiers ne contenant pas d'**Images de l'hôte ECI** (pour des fonctions non essentielles en matière de sécurité). L'**Opérateur** pourra jouer un rôle actif dans la gestion de la version logicielle sur l'**Équipement CPE.** Ce protocole facilite la coopération en fixant des normes de points d'interopérabilité technique entre le **Fabricant d'équipement CPE** et l'**Opérateur**:

- transfert standard volontaire des données téléchargées du **Fabricant d'équipement CPE** à l'**Opérateur**;

NOTE – Les détails techniques de ce transfert ne relèvent pas du champ d'application des spécifications relatives à l'interface **ECI**.

- protocole de transport via radiodiffusion standard (permettant de prévoir une lecture unique au niveau de la tête de réseau de radiodiffusion de l'**Opérateur**); et

- découverte standard, mise en œuvre du protocole de transport et choix des paramètres du protocole de transport opérationnel sur les équipements récepteurs.

Le Flux de transport de radiodiffusion de l'**Hôte ECI** et les mises en œuvre de l'**Équipement CPE** seront conformes à DVB SSU [ETSI TS 102 006] et de ce fait, à la section y afférente de la définition du carrousel de données DVB [ETSI EN 301 192], aux consignes de mise en œuvre [ETSI TR 101 202] et à la définition du carrousel de données MPEG [ISO/CEI 13818-6].

Les **Opérateurs** et les **Équipements CPE** prendront en charge le profil DVB-SSU simple et en option le profil DVB-SSU UNT.

Les **Opérateurs** pourront prendre en charge plusieurs carrousels simultanément.

Les **Équipements CPE** balaieront tous les carrousels signalés comme il se doit dans les informations relatives au système (SI), la table de notification des mises à jour (UNT) (le cas échéant) et la table de correspondance du programme (PMT) pour trouver les éléments à télécharger.

Le dispositif global de radiodiffusion relatif au téléchargement d'images est représenté dans la Figure 6.4.2.1-1.

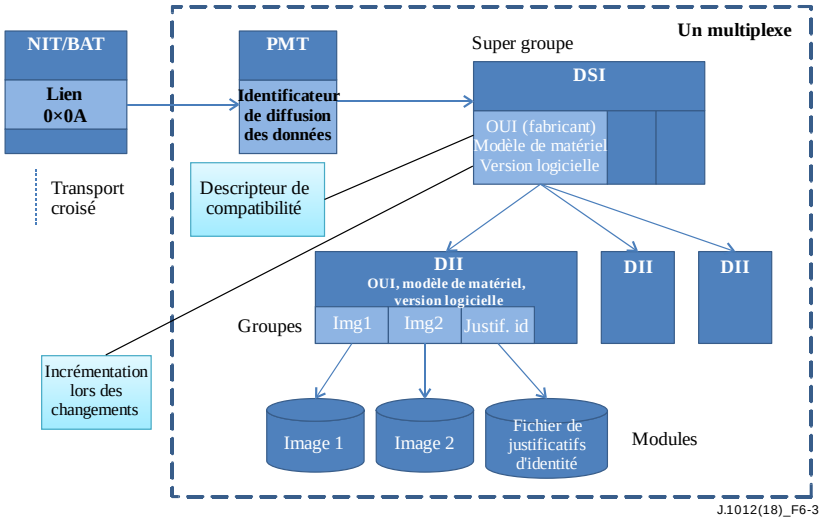


Figure 6.4.2.1-1 – Vue d'ensemble de la signalisation des images d'hôte et de la structure des carrousels (sans variante UNT)

6.4.2.2 Transfert du Fabricant d'équipement CPE à l'Opérateur

Tout **Écosystème ECI** futur devra définir des consignes à l'intention des **Opérateurs** et des **Fabricants d'Équipement CPE** visant à uniformiser la méthode d'échange des informations des fichiers d'images (**Images de l'hôte ECI** ou non), des justificatifs d'identité des images **ECI** et des métadonnées concernant le téléchargement à partir de (nombreux) **Fabricants d'Équipement CPE** vers de (nombreux) **Opérateurs**.

6.4.2.3 Signalisation DVB-SI

6.4.2.3.1 Signalisation de l'emplacement de téléchargement

Les **Opérateurs** prendront en charge le descripteur de lien DVB-SSU (type de lien 0x09) avec au moins l'identificateur unique d'organisation (OUI) DVB générique (à savoir le lien vers tous les carrousels non propre au **Fabricant**) dans toutes les tables NIT (réseaux terrestres ou câblés) ou BAT (satellite).

Les **Équipements CPE** au profil simple prendront en charge le descripteur de lien DVB-SSU (type de lien 0x09).

Les **Opérateurs** prenant en charge le profil DVB-SSU UNT seront compatibles avec le descripteur de lien de balayage SSU (type de lien 0xA) dans toutes les tables NIT (réseaux terrestres ou câblés) ou BAT (satellite).

Les **Équipements CPE** au profil UNT prendront en charge le descripteur de lien de balayage DVB-SSU (type de lien 0x09).

6.4.2.3.2 Mises à jour d'urgence

Afin d'indiquer la nécessité de remplacer d'urgence une **Image de l'Hôte ECI**, un ou plusieurs descripteurs ECI_host_emergency_download pourront être placés dans les tables NIT et BAT ou dans l'une des entrées de la table de description des services (SDT) pour les services auxquels l'**Hôte ECI** concerné peut fournir l'accès. L'**Hôte ECI** pourra extraire ce descripteur de l'une quelconque des tables dans lesquelles il apparaît dans les multiplexes en cours de syntonisation, exécuter le traitement associé et utiliser n'importe quel syntoniseur de rechange pour accéder aux multiplexes où il pourra acquérir ce descripteur en 30 minutes dans le pire des cas, pendant qu'il est sous tension. Il est recommandé de contrôler plus fréquemment les multiplexes sans syntoniseur (toutes les 3 minutes).

Le champ ECI_host_emergency_download_descriptor permet de cibler certaines plates-formes d'opérations ainsi que des **opérations de plate-forme** et des images de clients spécifiques afin de réduire le nombre d'**Utilisateurs** subissant une perturbation potentiellement imputable à des mises à jour d'urgence.

Lorsque l'**Hôte ECI** trouve un nouveau descripteur ECI_host_emergency_download, il compare sa configuration et celle du **Client ECI** aux informations de ciblage qui y figurent. S'il trouve une cible correspondante et si la version de l'image de l'hôte déjà installée requiert une mise à jour, l'**Hôte ECI** l'effectuera conformément à la valeur du champ emergency_indicator. Cette action perturbera les activités en cours des **Utilisateurs** faisant appel à l'**Équipement CPE.**

Le descripteur d'opération **ECI** est un descripteur DVB privé. Dans la table où il apparaît, il sera toujours précédé du champ DVB private_data_specifier_descriptor (voir [ETSI EN 300 468] et [ETSI TS 101 211]) utilisant le champ **ECI** private_data_specifier_field. La syntaxe du descripteur est définie dans le Tableau 6.4.2.3.2-1.

Tableau 6.4.2.3.2-1 – ECI_host_emergency_download_descriptor

Syntaxe	Nbre de bits	Mnémonique
ECI_host_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* main loop */		
main_loop_nr	8	uimsbf
for (i=0; i<main_loop_nr; i++){		
/* client loop */		
client_nr		
for (j=0; j<client_nr; j++){		
platform_operation_tag	8	uimsbf
Réservé	3	
client_flag	1	
client_tag	4	uimsbf
}		
/* host image loop */		
host_nr	8	uimsbf
for (j=0; j<host_nr; j++){		
Réservé	4	
emergency_indicator	4	uimsbf
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
min_host_version	8	uimsbf
}		
}		
/* private data till end of descriptor*/		
for (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Sémantique:

descriptor_tag	Valeur d'étiquette privée **ECI** du champ descriptor_tag: voir le document [b-UIT-T J Suppl. 7].
descriptor_length	Voir [b-ETSI EN 300 468].
main_loop_nr	Nombre d'entrées dans la boucle principale. Les entrées de boucle principale seront évaluées individuellement par l'**Hôte ECI**; en d'autres termes, elles auront une sémantique OR. Les divers éléments d'une entrée de boucle auront une sémantique AND.
client_nr	Nombre d'entrées dans la boucle cible du client. La valeur 0x00 signifiera que n'importe quel client correspondra. Chaque entrée de boucle possèdera une sémantique OR et tous les clients correspondants seront examinés afin de déterminer s'il faut procéder à une mise à jour d'urgence. Les champs d'une entrée de boucle auront une sémantique AND.
platform_operation_tag	Valeur d'étiquette d'**Opération de plate-forme ECI** figurant dans ECI_platform_operation_descriptor dans les tables NIT/BAT. L'**Hôte ECI** envisagera une mise à jour d'urgence si le champ platform_operation correspond au champ platform_operation de l'un des Clients installés.
client_flag	Indique si le champ client_tag peut permettre une mise en correspondance. La valeur 0b0 signifie non pertinent (à savoir, n'importe quel identificateur de client [client_id] correspondra). La valeur 0b1 signifie que le champ client_tag est pertinent.
host_tag	Valeur d'étiquette identifiant l'**Hôte ECI** figurant dans le champ ECI_platform_operation_descriptor des tables NIT/BAT correspondant au champ platform_operation_tag dans la même entrée de boucle de client. L'**Hôte ECI** envisagera une mise à jour d'urgence si les champs vendor_id et client_id correspondent à ceux des clients installés dans l'**Hôte ECI** pour l'**Opération de plate-forme**.
host_nr	Nombre d'entrées dans la boucle de l'hôte. La valeur minimale sera 1. Les entrées de boucle auront une sémantique OR: si l'une quelconque des spécifications de l'hôte correspond à la condition de cible, la boucle principale présente l'état correspondant.
emergency_indicator	L'**Hôte ECI** utilisera la valeur de ce champ pour sélectionner le comportement permettant de démarrer le téléchargement et la mise à jour subséquente de l'hôte définie dans le Tableau 6.4.2.3.2-2.
manufacturer_id	Identificateur de Fabricant de l'hôte cible d'une mise à jour d'urgence. L'**Hôte ECI** envisagera une mise à jour d'urgence si la valeur de ce champ correspond à l'identificateur de Fabricant (manufacturer_id) de l'**Hôte ECI**.
cpe_type_id	Valeur définie par le champ ECI_CPE_Type_ID dans le Tableau 6.2.2.1-2. L'**Hôte ECI** envisagera une mise à jour d'urgence si son champ cpe_type_id correspond à la valeur du présent champ. Une valeur 0x000 du champ cpe_type_id.cpe_type signifiera que le champ cpe_type de l'**Hôte ECI** correspond (et les champs cpe_model et host_version seront ignorés). Une valeur 0x00 du champ cpe_type_id.cpe_model signifiera que n'importe quel modèle d'**Équipement CPE** (cpe_model) d'**Hôte ECI** correspond (et la version de l'Hôte sera ignorée).
min_host_version	L'**Hôte ECI** envisagera une mise à jour d'urgence si, et seulement si, sa version est inférieure ou égale à la valeur de ce champ. NOTE – Une valeur de champ égale à 0xFF implique que toutes les versions de l'Hôte correspondent.
private_data_byte	Données privées: les contenus peuvent être définis par l'**Opérateur** qui gère la diffusion de ce descripteur.

Le Tableau 6.4.2.3.2-1 définit plusieurs conditions de la boucle principale (dotée d'une sémantique AND) à satisfaire pour que l'**Hôte ECI** envisage l'exécution d'une mise à jour d'urgence. Si toutes ces conditions sont remplies, l'**Hôte ECI** procédera à un téléchargement et à l'installation d'urgence d'une nouvelle Image d'hôte conformément au champ emergency_indicator de l'**Hôte ECI** concerné. Les valeurs du champ indicator sont définies dans le Tableau 6.4.2.3.2-2.

Tableau 6.4.2.3.2-2 – Valeurs des champs ECI_host_emergency_download_descriptor et emergency_indicator

Nom	Valeur	Description
System emergency	0x01	L'**Hôte ECI** téléchargera la nouvelle image d'hôte et l'installera le plus rapidement possible en interrompant les activités en cours initiées par l'**Utilisateur** si nécessaire. Voir la NOTE.
Urgence normale	0x03	L'**Hôte ECI** téléchargera la nouvelle image d'hôte et l'installera dès que possible sans perturber les activités initiées par l'**Utilisateur**. L'**Hôte ECI** téléchargera la nouvelle image d'hôte au plus tard lors de la prochaine mise sous tension. NOTE – Les **Opérateurs de plate-forme** peuvent s'en servir, par exemple si l'**Hôte ECI** actuel présente de graves défaillances en matière de services de déchiffrement mais peut fonctionner sans difficulté majeure dans les cas d'utilisation normale.
RFU	autre	Réservé à une utilisation future.
NOTE – Les **Opérateurs de plate-forme** peuvent s'en servir, par exemple si l'**Hôte ECI** actuel présente de graves problèmes de performance concernant les combinaisons cibles de plate-forme/client.		

6.4.2.4 Signalisation PSI

Les **Opérateurs** prendront en charge le champ data_broadcast_id_descriptor dans la table de correspondance du programme (PMT) [ETSI EN 300 468] pour chaque carrousel transmis mais n'ont pas à prendre en charge une quelconque signalisation d'identificateur unique d'organisation dans les octets de sélection de ce descripteur.

Les **Équipements CPE** au profil simple SSU utiliseront le champ data_broadcast_id_descriptor pour localiser l'identificateur de paquet (PID) du flux transportant un carrousel DVB-SSU.

6.4.2.5 Option UNT

Ce paragraphe ne s'applique qu'aux **Équipements CPE** et aux **Opérateurs** prenant en charge le profil UNT.

Dans la table de correspondance du programme, le champ data_broadcast_id_descriptor sera utilisé et contiendra la structure system_software_update_info avec le champ update_type 0x2 et le champ OUI défini sur DVB OUI 0x00015A.

Les **Opérateurs** effectueront une entrée dans l'une des tables SSU pour chaque type d'**Équipement CPE** qu'ils prennent en charge.

Les **Hôtes ECI** seront capables d'interpréter les descripteurs UNT suivants (voir [ETSI TS 102 006]):

- SSU_location_descriptor (si un carrousel est diffusé pour le type d'**Équipement CPE**).
- Scheduling_descriptor (si un carrousel est prévu dans un avenir proche pour le type d'**équipement CPE**).
- Message_descriptor (descripteur de messages).

Les **Équipements CPE** seront capables de procéder systématiquement au téléchargement réussi d'un carrousel reçu pratiquement sans erreur, monté et démonté aux heures publiées et effectuant deux cycles complets (répétition de tous les messages du carrousel) à condition qu'aucune activité initiée par un **Utilisateur** n'interfère avec le téléchargement.

6.4.2.6 Structure des carrousels

Les carrousels DVB SSU **ECI** (pour en savoir plus, voir [ETSI TS 102 006]) utiliseront des carrousels de données à deux couches.

Le carrousel DVB SSU **ECI** utilisera le message du lancement du serveur de téléchargement (DSI) avec les contraintes suivantes:

- Il existera une liste complète de tous les groupes disponibles pour le téléchargement.
- Chaque groupe correspondra à un modèle (**cpe_type + cpe_model)** d'un **Fabricant** et contiendra toutes les ressources nécessaires à l'**Hôte ECI** du type d'**Équipement CPE**. Cette

règle implique qu'un maximum de 255 modules (fichiers d'images) pourra être disponible (plus un fichier pour les justificatifs d'identité).

NOTE 1 – En raison de la limitation des valeurs du champ **ECI_host_id.model_id**, leur nombre ne pourra pas excéder 239.

- Le descripteur de compatibilité (CompatibilityDescriptor) du champ GroupCompatibility de la structure GroupInfoIndication (pour en savoir plus, voir [ETSI TS 102 006]) utilisera la convention suivante:
  - La boucle contiendra un descripteur de matériel système:
    - L'OUI correspondra au **Fabricant** de l'**Équipement CPE**.
    - Les champs de modèle et de version associés au descripteur du matériel système correspondront au type (**cpe_type**) et au modèle (**cpe_model**) de l'**Équipement CPE** et seront égaux aux champs **id.cp_type** et **id.cpe_model** du **Certificat de l'Hôte ECI** dans le fichier de justificatifs d'identité du groupe.
  - La boucle contiendra un descripteur de logiciel système. Le champ "model" sera mis à 0, le champ "version" contiendra la version de tous les logiciels d'**Hôte ECI** du groupe (autrement dit, les **Images de l'hôte ECI** et les **images** qui n'y sont pas associées).

Les **Équipements CPE** utiliseront les champs de modèle et de version du CompatibilityDescriptor correspondant au modèle et à la version des **Équipements CPE** ainsi que le champ de version du logiciel pour vérifier si le groupe contient une mise à jour; auquel cas, ils téléchargeront les nouvelles images.

Le carrousel DVB SSU **ECI** utilisera les champs de message de l'infrastructure d'identité numérique (DII) avec les contraintes suivantes:

- La taille de bloc (BlockSize) sera mise à 2 kilooctets (2 048 octets) minimum.
- Le champ "tDownloadScenario" recevra une valeur significative correspondant au téléchargement de tous les modules avec une durée de répétition égale à au moins 4 fois le message le plus lent (durée du délai d'exécution du carrousel).
- Le champ "moduleId bits 7..0" sera égal au champ **id.image_model** du fichier d'images.
- Le champ "moduleVersion" sera égal au champ **ECI id.image_version** du fichier d'images.

Les **Équipements CPE** pourront utiliser le champ "tDownloadScenario" pour interrompre les téléchargements qui échouent (par exemple, en raison de taux élevés d'erreurs de paquets) et signaler le problème à l'**Utilisateur.**

Le groupe d'un type d'**Équipements CPE** contiendra les modules suivants:

- Fichiers d'images pour un type d'**Équipement CPE** (il pourra s'agir d'un ensemble d'images partiel).
- Le fichier de justificatifs d'identité des **Images de l'hôte ECI** contenant les justificatifs (les plus récents) de toutes les images d'un **Hôte ECI**:
  - le champ "DII moduleId bits 7..0" de ce module sera mis à 0xFF; et
  - le champ "moduleVersion" augmentera à chaque changement.

NOTE 2 – Entre les téléchargements, les **Opérateurs** ont le droit de partager les fichiers communs à divers types d'**Équipements CPE** en partageant les blocs de données de téléchargement (DownloadDataBlocks) entre les infrastructures d'identité numérique. Cela nécessite toutefois une gestion cohérente des identificateurs d'**Images de l'hôte ECI** entre les types d'**Équipements CPE**.

6.4.2.7 Opération de téléchargement des Hôtes ECI

Le chargeur d'**Image de l'Hôte ECI** s'efforcera de vérifier tous les carrousels possibles toutes les 30 minutes à l'état sous tension si des ressources d'accès au réseau sont disponibles et au moins toutes

les 6 heures en état de veille sans perturber l'**Utilisateur**, par exemple, après la mise en veille de l'**Équipement CPE** et pendant les heures de faible audience.

Si un fournisseur de réseau met à disposition des UNT transportant des téléchargements potentiels pour un type d'**Équipement CPE**, l'équipement correspondant les vérifiera régulièrement afin de savoir si une nouvelle mise à jour est programmée. L'**Équipement CPE** effectuera ce contrôle aux mêmes conditions de fréquence que pour les carrousels d'**Images de l'hôte ECI**.

Il est conseillé d'avertir l'**Utilisateur** si un **Équipement CPE** en mode exclusivement radiodiffusion n'est pas en mesure d'effectuer ces contrôles pendant plus de deux semaines.

Une fois détectée la disponibilité d'un nouveau téléchargement, ce qui signifie que l'**Équipement CPE** et l'**Utilisateur** ont donné leur approbation, l'**Équipement CPE** s'efforcera d'effectuer le téléchargement et d'installer la nouvelle image (éventuellement en écrasant une version antérieure). La persistance de l'échec du téléchargement sera signalée à l'**Utilisateur** selon la méthode appropriée. Les **Hôtes ECI** seront toujours capables de se rétablir après l'échec du téléchargement d'une image d'hôte et de retrouver un état fonctionnel, par exemple, en restaurant l'image d'hôte antérieure ou en tentant de charger à nouveau la nouvelle image d'hôte.

À noter que la persistance de l'échec du téléchargement de nouvelles **Images de l'hôte ECI** ou de justificatifs d'identité peut entraîner un déni de service de la part de l'**Opérateur**.

6.4.2.8 Programmation des carrousels des Opérateurs

Les **Opérateurs** doivent fournir une largeur de bande suffisante pour que les carrousels de données d'images des **Équipements CPE** effectuent le téléchargement dans un délai raisonnable.

6.4.2.9 Aspects relatifs à l'interface d'Utilisateur

Les **Équipements CPE** capables de télécharger des **Images de l'hôte ECI** sur le réseau de radiodiffusion disposeront:

- d'un mode de balayage du téléchargement qui automatisera les contrôles de disponibilité de nouvelles images ou justificatifs d'identité à intervalle régulier, par exemple pendant l'état de veille. Il est recommandé au **Fabricant** de définir ce mode comme paramètre par défaut du contrôle des téléchargements;
- d'une option dans le menu de l'**Équipement CPE** qui automatisera l'approbation par l'**Utilisateur** des nouveaux fichiers ou justificatifs d'identité d'**Images de l'hôte ECI**. Il est recommandé au **Fabricant** de définir cette option comme paramètre par défaut de l'approbation des téléchargements.

Les **Équipements CPE** prévoiront au moins une autre méthode de téléchargement des nouveaux fichiers d'**images d'Hôte ECI** afin d'éviter que ceux qui fonctionnent sur des réseaux de radiodiffusion ne fournissant pas de nouveaux fichiers d'**images d'Hôte ECI** pour leur type d'équipement fassent l'objet d'un déni de service.

6.4.3 Protocole de transport des Hôtes ECI sur Internet

6.4.3.1 Protocole IP

L'interface **ECI** ne définit pas de protocole particulier de contrôle par l'**Équipement CPE** de nouveaux fichiers d'**Images de l'hôte ECI** issus d'un service fourni par le **Fabricant.** Il est néanmoins recommandé d'utiliser le protocole de transfert de fichiers HTTP1.1 [IETF RFC 7231] ainsi qu'éventuellement le protocole indiqué au § 7.7.3.3, qui définit un service de téléchargement normalisé pour les fichiers d'**Images de l'hôte ECI** à partir du serveur d'une **Opération de plate-forme**.

En général, le serveur de téléchargement des **Images de l'hôte ECI** est prévu par le **Fabricant de l'Équipement CPE**. En cas de dispositions particulières entre le **Fabricant de l'Équipement CPE**

et un **Opérateur** (ou des tiers agissant pour leur compte), celles-ci peuvent également être fournies par l'**Opérateur** ou un tiers.

6.4.3.2 Opération de chargement en ligne

Le chargeur **ECI** d'**Images de l'hôte ECI** en ligne s'efforcera de vérifier son serveur en ligne toutes les 30 minutes sans gêner l'**Utilisateur**. Il est conseillé d'avertir l'**Utilisateur** si un **Équipement CPE** en mode en ligne seulement ne peut pas effectuer ces contrôles pendant plus longtemps.

Après détection de la disponibilité d'un nouveau téléchargement, l'**Équipement CPE** s'efforcera d'effectuer ce dernier, et d'installer la nouvelle image (éventuellement en écrasant des versions d'images antérieures). La persistance de l'échec du téléchargement sera signalée à l'**Utilisateur** selon la méthode appropriée.

À noter que l'échec du téléchargement de nouvelles **Images de l'hôte ECI** ou justificatifs d'identité pourra entraîner un déni de service de la part de l'**Opérateur**.

Le chargeur en ligne de l'**Équipement CPE** fournira un ensemble de (nouvelles) images et de (nouveaux) justificatifs d'identité d'images comme défini au § 6.3 à des fins de vérification, de stockage et d'activation.

Le chargeur **ECI** d'**Images de l'hôte** en ligne possèdera des fonctions de téléchargement d'urgence dont les effets seront identiques à ceux définis au § 6.4.2.3.2 pour la radiodiffusion.

6.4.4 Autres protocoles de transport

L'**Hôte ECI** est autorisé à utiliser d'autres protocoles de livraison (propriétaires).

Le chargeur de l'**Équipement CPE** traitera un ensemble de (nouvelles) images et de (nouveaux) justificatifs d'identité d'images comme défini au § 6.3 à des fins de vérification, de stockage et d'activation.

7 Chargeur de Client ECI

7.1 Introduction

L'**Hôte ECI** peut télécharger, stocker et activer des **images de Clients ECI** et les données associées. Le processus de chargement des **Clients ECI** comporte les étapes suivantes:

1) Découverte de la protection basée sur l'interface **ECI** d'un service/bouquet de services ou autres méthodes déterminant la nécessité d'installer d'un **Client ECI**. Ce processus fait partie de l'application de navigation normale de l'**Équipement CPE**.
2) Détermination de l'emplacement du réseau (radiodiffusion ou en ligne) où se trouvent les ressources requises pour installer un **Client ECI** sur l'**Hôte ECI**.
3) Téléchargement et stockage (dans la mémoire non volatile) des informations de l'**Opération de plate-forme** requises pour installer le **Client ECI** et en vérifier les justificatifs d'identité.
4) Enregistrement de l'**Hôte ECI** dans le système de sécurité de l'**Opération de plate-forme** et réception (si nécessaire) des données d'initialisation propres à l'**Équipement CPE** afin de procéder au déchiffrement du **Client ECI**.
5) Téléchargement et stockage (dans la mémoire non volatile) de l'**Image de client ECI** et des justificatifs d'identité du **Client ECI** associés à partir du réseau, vérification des justificatifs d'identité et de l'image et stockage dans la mémoire non volatile en vue d'une utilisation future.
6) Initialisation du **Client ECI** à l'aide de l'**Image de client ECI**, du **Certificat d'opération de plate-forme,** de l'attribution d'un Conteneur **ECI** et des ressources de sécurité évoluée requises et démarrage de l'exécution du **Client ECI**.

Tous les processus peuvent être effectués à l'aide de données provenant du flux de radiodiffusion ou d'Internet, à l'exception de l'enregistrement de l'**Équipement CPE** auprès de l'**Opérateur**, qui requiert une assistance manuelle au cas où seule une connexion par radiodiffusion serait disponible.

Les **Opérateurs** peuvent renouveler les ressources des **Clients ECI** à tout moment en publiant les informations sur les réseaux de radiodiffusion ou en ligne. L'**Hôte ECI** vérifie régulièrement la présence de ces mises à jour.

L'interface **ECI** nécessite des données de support pour diverses fonctions des **Équipements CPE**, telles que des données de révocation ou des **Chaînes de Certificats** à jour dont le **Client ECI** et/ou l'**Hôte ECI** ont besoin pour prendre en charge le **Client ECI.** Sur les réseaux de radiodiffusion, le protocole de transport permet le téléchargement sélectif des données requises par un **Équipement CPE** sur la base d'un "indice" (hachage) de l'identification des données. Le regroupement des données par le hachage de l'indice s'appelle la "mise en seau" (bucketizing). Sur les réseaux en ligne, le téléchargement sélectif repose sur la transmission de l'identification des données requises sous forme d'un paramètre envoyé à une API de services web.

L'**Hôte ECI** peut télécharger les éléments de données suivants:

- I**mages de Clients ECI** (au format "seau" sur les réseaux de radiodiffusion).
- Données de révocation des **Clients ECI** (au format "seau" sur les réseaux de radiodiffusion).
- Chaîne de clients d'opération de plate-forme.
- Données de révocation des **opérations de plate-forme** (au format "seau" sur les réseaux de radiodiffusion).
- Données de révocation des **images d'hôtes ECI** (au format "seau" sur les réseaux de radiodiffusion). Données d'initialisation des clients de configuration de la sécurité évoluée **ECI** pour le déchiffrement des images de clients chiffrées (au format "seau" sur les réseaux de radiodiffusion).

7.2 Découverte des Clients ECI

7.2.1 Introduction

En général, un **Équipement CPE** conforme **ECI** (par exemple, une iDTV) n'aura pas de **Client ECI** installé à son départ de l'usine parce qu'il pourra être vendu sur n'importe quel marché dans le monde. Le paragraphe suivant définit les mécanismes permettant à un **Équipement CPE** conforme **ECI** de trouver les **Clients ECI** éventuellement requis pour désembrouiller les services fournis par un réseau auquel il est connecté.

Le processus de découverte distingue deux types de réseaux:

1) réseaux basés sur des flux de transport (radiodiffusion et réseaux TVIP types);
2) réseaux basés sur le protocole IP.

Dans le cas des réseaux basés sur des flux de transport, l'interface **ECI** prend en charge deux modes de découverte des fournisseurs et des clients:

1) Installation manuelle – y compris les paramètres de configuration de base du réseau (radiodiffusion).
2) Autonome (avec choix de l'**Utilisateur**) – suppose que l'**Équipement CPE** peut s'installer sur le réseau de manière autonome.

Les protocoles d'installation manuelle et autonome sur les réseaux basés sur des flux de transport utilisent une signalisation commune.

Dans le cas des réseaux basés sur le protocole IP, l'interface **ECI** prend en charge l'entrée manuelle de l'adresse URL de base.

7.2.2 Réseaux basés sur des flux de transport

7.2.2.1 Signalisation commune

Afin de réduire le nombre de paramètres à entrer manuellement par l'**Utilisateur**, l'interface **ECI** fournit une signalisation en ligne des principaux paramètres **ECI** d'installation des clients:

- Un ou plusieurs descripteurs ECI_platform_operation_descriptors dans la table NIT contenant les clients disponibles (par identificateur) par **Opération de plate-forme**. Le descripteur comprend le nom du fournisseur de la plate-forme et un identificateur court (afin d'autoriser sa représentation compacte dans la chaîne d'installation manuelle).
- Le fournisseur de plate-forme peut spécifier une adresse URL de base pour l'API web dans le champ ECI_base_URL_descriptor.

7.2.2.2 ECI_ platform_operation_descriptor

Le champ ECI_ platform_operation_descriptor fournit des informations essentielles sur l'**Opération de plate-forme** proposant des services d'accès à un réseau basé sur des flux de transport.

Pour chaque **Opération de plate-forme,** la table NIT_{actual} (et/ou BAT sur les réseaux satellite) contiendra le descripteur ECI_ platform_operation_descriptor au minimum dans le multiplexe central et la table identifiés dans la chaîne d'installation des réseaux ne proposant qu'une installation manuelle et dans tous les multiplexes, hormis pour les réseaux satellite, pour les réseaux prenant en charge le mode autonome. Les réseaux satellite ne sont autorisés à transporter le descripteur ECI_ platform_operation_descriptor que sur les multiplexes où le fournisseur propose des services: soit dans la table NIT, soit dans une table BAT.

ECI_ platform_operation_descriptor est un descripteur privé DVB utilisant le spécificateur de données privées de l'interface **ECI** dans le champ DVB private_data_specifier_descriptor [ETSI TS 101 162]. Il est défini dans le Tableau 7.2.2.2-1.

Tableau 7.2.2.2-1 – ECI_ platform_operation_descriptor

Syntaxe	Nbre de bits	Mnémonique
ECI_ platform_operation_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
platform_tag	8	uimsbf
operator_id	20	uimsbf
platform_operation_id	20	uimsbf
platform_name_length	8	uimsbf
/* platform name loop */		
for (i=0; i<N; i++){		
platform_name_char	8	uimsbf
}		
for (i=0; i<N; i++){		
extension_byte	8	uimsbf
}		
}		

Sémantique:

descriptor_tag	Valeur d'étiquette privée **ECI** du champ descriptor_tag. Voir le document [b-UIT-T J Suppl. 7].
platform _tag	Ce champ 8 bits spécifie l'étiquette de l'**Opération de plate-forme** aux fins d'installation manuelle. Chaque table NIT ou BAT d'un réseau prenant en charge chaque **Opération de plate-forme** aura une valeur platform_tag unique. Ce champ ne figurera qu'une seule fois dans chaque table NIT ou BAT. Il ne servira pas à ordonnancer les fournisseurs et ne sera pas présenté dans l'interface d'**Utilisateur** d'un **Équipement CPE** pour sélectionner une **Opération de plate-forme**.
operator_id	Identificateur d'opérateur défini au § 7.5.2 de la présente Recommandation. Il identifie l'**Opérateur** de l'**Opération de plate-forme**.
platform_operation_id	Identificateur d'**Opération de plate-forme** défini au § 7.5.3 de la présente Recommandation.
platform_name_length	Longueur de la séquence d'octets de la boucle du nom de plate-forme. Si elle est égale à 0, le fournisseur ne prendra pas en charge le mode autonome et ne figurera pas dans le menu de sélection des fournisseurs du menu d'installation des clients de l'**Équipement CPE**. La valeur maximale de ce champ sera de 40.
platform_name_char	Séquence de caractères UTF8 représentant le nom de l'Opération de plate-forme.
extension_byte	Octets supplémentaires réservés à une utilisation future par la présente Recommandation.

7.2.2.3 ECI_base_url_descriptor

Le champ ECI_base_url_descriptor permet à l'**Opération de plate-forme** de signaler l'adresse URL de base de son API web (voir le § 7.7.3) qui peut servir à fournir des services en rapport avec l'installation des clients en cas d'accès en ligne.

Pour chaque **Opération de plate-forme**, la table NIT_{actual} (et/ou BAT sur les réseaux satellite) pourra contenir le champ ECI_base_url_descriptor dans la table contenant le champ ECI_platform_operation_descriptor.

ECI_base_url_descriptor est un descripteur privé DVB utilisant le spécificateur de données privées de l'interface **ECI** dans le champ DVB private_data_specifier_descriptor [ETSI EN 300 468]. Il est défini dans le Tableau 7.2.2.3-1.

Tableau 7.2.2.3-1 – ECI_base_url_descriptor

Syntaxe	Nbre de bits	Mnémonique
`ECI_base_url_descriptor(){`		
`descriptor_tag`	8	uimsbf
`descriptor_length`	8	uimsbf
`platform_tag`	4	uimsbf
`réservé`	4	
`base_url_length`	8	uimsbf
`/* base url loop */`		
`for (i=0; i<N; i++){`		
`base_url_char`	8	uimsbf
`}`		
`}`		

Sémantique:

descriptor_tag	Valeur d'étiquette privée **ECI** du champ descriptor_tag. Voir le document [b-UIT-T J Suppl. 7].
platform_tag	Ce champ de 4 bits spécifie l'étiquette du fournisseur aux fins d'installation manuelle. Dans chaque table NIT ou BAT d'un réseau le prenant en charge, chaque **Opération de plate-forme** présentera une valeur platform_tag unique. Ce champ ne figurera qu'une seule fois dans chaque table NIT ou BAT. Il ne servira pas à ordonnancer des **opérations de plate-forme** et ne sera pas présenté dans l'interface d'**Utilisateur** d'un **Équipement CPE** pour sélectionner une **Opération de plate-forme**.
base_url_length	Ce champ indiquera le nombre d'octets de la boucle de l'adresse URL de base.
base_url_char	Séquence de caractères UTF8 constituant l'adresse URL de base d'une Opération de plate-forme.

7.2.2.4 Installation manuelle

L'**Opération de plate-forme** peut fournir une chaîne d'installation à l'**Utilisateur**, que celui-ci pourra entrer dans une option de menu d'installation adaptée de l'interface d'**Utilisateur** d'un **Équipement CPE** dans le but d'installer un **Client ECI**. La définition de la chaîne d'installation sera conforme au présent paragraphe. La chaîne d'installation est composée de chiffres représentant un nombre binaire de longueur variable. Dans une représentation à bit de plus fort poids en premier, le nombre binaire peut être construit en concaténant les valeurs binaires 3 bits des chiffres d'une représentation à bit de plus fort poids en premier.

Le nombre est présenté à l'**Utilisateur** en fragments de 4 chiffres et l'entrée dans l'interface d'utilisateur de l'**Équipement CPE** représentera également des fragments de 4 chiffres.

Le Tableau 7.2.2.4-1 définit les paramètres de la chaîne d'installation.

Tableau 7.2.2.4-1 – Paramètres de la chaîne d'installation (en nombre de bits)

Paramètre	DVB-T/DVB-T2	DVB-C/DVB-C2	DVB-S/DVB-S2	TVIP	Mnémonique
Type de réseau	3	3	3	3	uimsbf
Identificateur de réseau	16	17	17	16	uimsbf
Étiquette de plate-forme	8	8	8	8	uimsbf
Identificateur de Client	4	4	4	4	uimsbf
Bourrage	0	0	0	0	uimsbf
Somme de contrôle	5	5	5	5	uimsbf
Nombre de bits	36	36	36	36	uimsbf
Nombre de chiffres	12	12	12	12	uimsbf
Nombre de fragments	3	3	3	3	uimsbf

Sémantique:

Type de réseau	Champ de 3 bits. Les valeurs du type de réseau sont présentées dans le Tableau 7.2.2.4-2.
Identificateur de réseau	Identificateur de table DVB SI contenant le champ ECI_service_provider_descriptor (voir le § 7.2.2.2) fournissant les informations détaillées requises pour accéder aux services définis dans le Tableau 7.2.2.4-3.
Étiquette de plate-forme	Champ de 4 bits représentant l'étiquette de fournisseur du fournisseur de services requis dans le champ ECI_service_provider_descriptor de la table NIT ou BAT.
Identificateur de Client	Champ de 4 bits représentant l'étiquette de fournisseur du client requis dans le champ ECI_service_provider_descriptor sélectionné par l'étiquette de fournisseur dans la table NIT ou BAT.
Bourrage	Champ de 0...2 bits à valeur nulle rembourrant la chaîne précédente afin d'obtenir un multiple de 3 bits.
Somme de contrôle	Champ de 5 bits constitué par ajout de fragments successifs de 5 bits de la chaîne précédente. La dernière partie de la chaîne est rembourrée par des zéros de tête supplémentaires afin d'atteindre une longueur de 5 bits. Par exemple, la somme de contrôle de la chaîne 0b01011010 est 0b01011 + 0x00010 = 0b01101. La somme de contrôle sera utilisée par l'interface d'**Utilisateur** de l'**Équipement CPE** pour rejeter les entrées erronées de l'**Utilisateur**.

Tableau 7.2.2.4-2 – Représentation des valeurs de type de réseau

Type de réseau	Valeur
DVB-T/T2	0
DVB-C/C2	1
DVB-S/S2	2
TVIP	3
Réservé	4..7

Tableau 7.2.2.4-3 – Représentation de l'identificateur de réseau

Type de réseau	Valeur de l'identificateur de réseau	Nombre de bits
DVB-C	0b0 suivi de l'identificateur de réseau de la table NIT ou 0b1 suivi de l'identificateur de BAT de la table BAT.	17
DVB-S/S2	0b0 suivi de l'identificateur de réseau de la table NIT ou 0b1 suivi de l'identificateur de BAT de la table BAT.	17 17

7.2.2.5 Installation en mode autonome

Le recours à ce mode d'installation requiert que l'**Équipement CPE** soit capable de découvrir lui-même les paramètres du réseau basé sur des flux de transport et donc d'accéder à tous les flux de transport du réseau.

Chaque service de chaque multiplexe sera associé à l'**étiquette d'opérations de plate-forme ECI** à même de donner accès au service. Cela peut être fait dans la table de description des services (SDT) en fonction du service (voir le § 7.2.2.6) ou dans la table NIT ou BAT (uniquement pour les réseaux satellite) en fonction du multiplexe (voir le § 7.2.2.6).

L'**Équipement CPE** disposera d'une option permettant à l'**Utilisateur** d'installer n'importe quel **Client ECI** des **opérations de plate-forme** dans le cadre du processus d'installation autonome. Si l'**Utilisateur** décide d'installer un **Client ECI** de l'**Opération de plate-forme** afin de recevoir des services déchiffrés via le réseau d'accès apparenté, le comportement par défaut de l'**Équipement CPE** consistera à installer tous les **Services** étiquetés pour cette **Opération de plate-forme** dans la liste centralisée des services de l'**Équipement CPE**.

7.2.2.6 Descripteur de l'étiquette de services ECI

Le champ ECI_service_tag_descriptor est contenu dans la table SDT. Il associe à chaque service les fournisseurs de services **ECI** qui en proposent le désembrouillage. Il est défini dans le Tableau 7.2.2.6-1.

Tableau 7.2.2.6-1 – Descripteur de l'étiquette de services ECI

Syntaxe	Nombre de bits	Mnémonique
`ECI_service_tag_descriptor(){`		
`descriptor_tag`	8	uimsbf
`descriptor_length`	8	uimsbf
`platform_tag`	8	uimsbf
`}`		

Sémantique:

descriptor_tag	Valeur d'étiquette privée **ECI** du champ descriptor_tag. Voir le document [b-UIT-T J Suppl. 7].
platform_tag	Valeur du champ platform_tag de l'**Opération de plate-forme ECI** figurant dans le champ ECI_platform_operation_descriptor et contenue dans la table NIT ou la table BAT du réseau.

7.2.2.7 Descripteur de liste de plates-formes ECI

Le descripteur de liste de plates-formes **ECI** fournit la liste des **opérations de plate-forme ECI** donnant accès aux services des différents multiplexes du réseau. Le champ

ECI_platform_list_descriptor est contenu dans la table NIT et/ou BAT. Il est défini dans le Tableau 7.2.2.7-1.

Tableau 7.2.2.7-1 – ECI_platform_list_descriptor

Syntaxe	Nombre de bits	Mnémonique
ECI_platform_list_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
platform_count	8	uimsbf
/* platform loop */		
for (j=1; j<M; j++){		
platform_tag	8	uimsbf
}		
service_count	16	uimsbf
/* service loop */		
for (j=0; j<M; j++){		
service_id	16	uimsbf
}		
}		
}		

Sémantique:

descriptor_tag	Valeur d'étiquette privée ECI du champ descriptor_tag. Voir le document [b-UIT-T J Suppl. 7].
platform_count	Ce champ de 8 bits correspond au nombre d'étiquettes de fournisseurs dans la boucle suivante.
platform_tag	Valeur du champ platform_tag de l'Opération de plate-forme ECI figurant dans le champ ECI_platform_operation_descriptor et contenue dans la table NIT ou BAT du réseau. Les services auxquels l'Opération de plate-forme étiquetée est associée suivent dans la boucle des services. Les valeurs du champ platform_tag peuvent apparaître plusieurs fois dans la boucle externe de ce descripteur.
service_count	Champ de 16 bits représentant le nombre d'identificateurs de services dans la boucle suivante.
service_id	Identificateur de services DVB d'un service dans le multiplexe de la table NIT ou BAT auquel il est possible d'accéder avec les services d'accès des plates-formes auxquelles fait référence la boucle de plates-formes précédente.

7.2.3 Découverte des clients fondée sur le réseau IP

7.2.3.1 Installation manuelle

Un **Équipement CPE** doté d'un accès aux réseaux IP proposera une option d'entrée manuelle des adresses URL afin de rendre possible l'installation d'un prestataire de services. L'adresse URL servira d'adresse URL de base de l'API web.

NOTE – Dans le cadre de leurs fonctions d'application, dont certaines sont téléchargeables, les **Équipements CPE** pourront proposer l'accès à divers services en ligne. Ils pourront suggérer un fournisseur de services et une interface API d'installation des clients afin d'automatiser le processus d'installation des clients pour l'**Utilisateur**.

7.2.3.2 Installation fondée sur une page web

Ce type d'installation d'un **Client ECI** ne relève pas de la présente Recommandation et peut faire l'objet de spécifications supplémentaires.

7.3 Stockage, vérification et activation

7.3.1 Politiques générales en matière de mises à jour

L'interface **ECI** prend en charge le renouvellement fréquent des éléments afin de renforcer l'intégrité. De ce fait, l'existence de mises à jour est contrôlée à intervalle régulier pour tous les éléments téléchargés. La politique suivante en matière de téléchargement des mises à jour s'appliquera à tous

les **Clients ECI** ainsi qu'aux données des **opérations de plate-forme** et aux données de révocation associées.

Les **Hôtes ECI** s'efforceront de vérifier l'existence de mises à jour à intervalle régulier et informeront l'**Utilisateur** s'il convient d'effectuer une quelconque action. Des exigences détaillées concernant la politique en matière de mises à jour sont proposées dans le document [b-UIT-T J Suppl. 7].

L'**Hôte ECI** stockera les **Chaînes de clients d'opérations de plate-forme** avec le **Client ECI** associé. Le stockage et la suppression seront gérés dans le cadre de l'installation et de la suppression des **Clients ECI**.

L'**Hôte ECI** mettra automatiquement à jour le **Certificat** du fournisseur de plate-forme et écrasera les versions antérieures.

7.3.2 Téléchargement et stockage des images de Clients ECI

Dans le cadre de la gestion des ressources liées aux **Clients ECI**, l'**Hôte ECI** stockera l'**Image de client ECI** requise pour accéder aux services ou au contenu de la mémoire non volatile uniquement après l'accord (implicite) de l'**Utilisateur**. Toute politique automatique d'installation de **Clients ECI** mettra à la disposition de l'**Utilisateur** une méthode transparente de traitement de l'éventuelle limitation des ressources afin de gérer les **Clients ECI** de manière transparente pour l'**Utilisateur** sans perte inopinée d'accès aux contenus ou aux services. De ce fait, la suppression d'une **Image de client ECI** sera (implicitement) approuvée par l'**Utilisateur**.

L'**Hôte ECI** stockera les **Clients ECI** téléchargés dans la mémoire non volatile avec leurs justificatifs d'identité d'origine sur la base d'une **Opération de plate-forme.** Les nouvelles versions des **Clients ECI** (ne comprenant que les nouveaux justificatifs d'identité) écraseront les versions antérieures (sur la base d'une **Opération de plate-forme**). Exemple: si deux **opérations de plate-forme** utilisent différentes versions d'un même type de **Client ECI**, l'**Hôte ECI** stockera les deux versions.

Une taille d'image minimale qu'un **Équipement CPE** peut stocker par créneau de **Client ECI** est proposée dans le document [b-UIT-T J Suppl. 7].

7.3.3 Validation et activation des Clients ECI

L'**Hôte ECI** chargera la **Chaîne de clients d'opération de plate-forme** la plus récente (sur la base du numéro de version) du **Certificat d'opération de plate-forme** dans le **Système de sécurité évoluée** et s'efforcera d'installer la clé publique de l'**Opération de plate-forme**, conformément aux règles génériques de traitement des chaînes définies au § 5.4.2.

L'**Hôte ECI** chargera le **Client ECI** le plus récent dans le **Système de sécurité évoluée** et la cosignature du client de l'**Opération de plate-forme** dans le **Système de sécurité évoluée**. Il validera ensuite le **Client ECI** conformément aux règles génériques de traitement des chaînes exposées dans le § 5.5 et vérifiera la signature et la cosignature de l'**Image de client ECI**. L'**Hôte ECI** notifiera l'**Utilisateur** en cas de révocation.

L'installation et l'activation d'un nouveau **Client ECI** dépendront de la réussite du processus de validation.

7.4 Formats de la structure des chaînes de Clients ECI

7.4.1 Introduction aux formats de la structure des chaînes de Clients ECI

La Figure 7.4.1-1 représente la structure de la **Chaîne de certificats de Clients ECI**. La chaîne commence par la **Liste de révocation** des fournisseurs, suivie du **Certificat du fournisseur** de **systèmes de sécurité** et de la **Liste de révocation des Clients ECI.** Elle finit par le fichier d'**Image de client ECI**. S'il s'agit d'une **Série d'images**, un **Certificat d'image de Client ECI** supplémentaire est introduit. La signature du **Client d'opération de plate-forme ECI** confère une seconde signature

à l'image du client permettant d'appliquer le **Client ECI** à une opération de plate-forme. Ce point est défini dans le § 7.5.

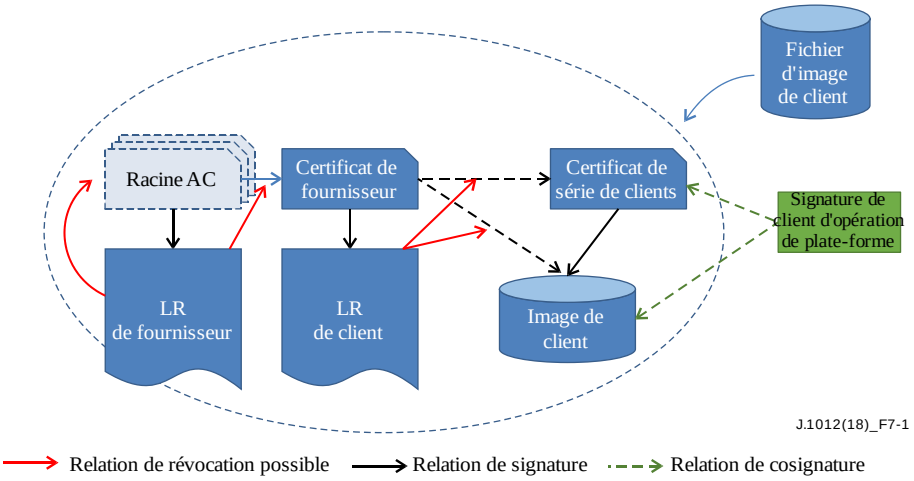


Figure 7.4.1-1 – Chaîne d'authentification des clients

7.4.2 Certificats de fournisseurs de systèmes de sécurité

Les **Certificats de fournisseurs** de **systèmes de sécurité** sont définis par la structure ECI_certificate. L'identificateur du **Certificat de fournisseur de systèmes de sécurité** est défini dans le Tableau 7.4.2-1.

Tableau 7.4.2-1 – Définition de l'identificateur de fournisseur de systèmes de sécurité

Syntaxe	Nbre de bits	Mnémonique
`ECI_Vendor_Id {`		
`padding(4)`		
`type` `/* voir le Tableau 5.2-2 */`	4	uimsbf
`vendor_id`	20	uimsbf
`vendor_version`	8	uimsbf
`}`		

Sémantique:

type: nombre entier	Valeur conforme au Tableau 5.2-2.
vendor_id: nombre entier	Numéro attribué au Fournisseur de systèmes de sécurité, unique dans le contexte de l'interface **ECI**.
vendor_version: nombre entier	Identificateur attribué de manière incrémentielle à la version du **Certificat** du Fournisseur de systèmes de sécurité. Les valeurs 0x00 et 0xF0..0xFF sont réservées.

7.4.3 Identificateurs de Certificat de série de Clients ECI et de série cible

Les **Certificats** de série de **Clients ECI** sont définis par la structure **ECI_Certificate**. L'identificateur du **Certificat de fournisseur de systèmes de sécurité** est défini dans le Tableau 7.4.3-1.

Tableau 7.4.3-1 – Définition de l'identificateur de série de clients

Syntaxe	Nbre de bits	Mnémonique
`ECI_Client_Series_Id {`		
`padding(4)`		
`type` `/* voir le Tableau 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Sémantique:

type: nombre entier	Valeur conforme au Tableau 5.2-2.
client_type: nombre entier	Type de **Client ECI** unique dans le contexte de l'identificateur de fournisseur de systèmes de sécurité de **Client ECI**.
client_version_major: integer	Numéro de version majeure du **Client ECI** d'un **type de Client ECI**. Les versions augmentent de façon incrémentielle en cas de nouvelle version majeure (voir la NOTE).
client_version_minor: integer	Numéro de version mineure du **Client ECI**. La comparaison avec un numéro de version mineure dans les **Listes de révocation de Clients ECI** peut entraîner la révocation et le remplacement automatique de ces clients.
NOTE – Le remplacement d'un **Client ECI** en cas de publication d'une version majeure n'est pas automatique dans les **Équipements CPE** conformes **ECI** car seules les mises à jour de versions mineures sont déclenchées automatiquement.	

NOTE – Les **Certificats** de séries de types de **Clients ECI** sont attribués aux **Clients ECI** nécessitant des mises en œuvre personnalisées propres à un **Équipement CPE** qui sont identiques en matière de sécurité et de fonctionnement.

L'identificateur de client cible est défini de la même manière que celui des **Hôtes ECI** à l'aide de la structure ECI_Host_Series_Image_Target_Id. Il lie une image de client à un **Hôte ECI** particulier.

7.4.4 Signature d'Image de client ECI

Les signatures de **Clients ECI** utiliseront la structure ECI_Data_Signature définie au § 5.6.

L'identificateur de **Client ECI** est défini dans le Tableau 7.4.4-1 et présente une structure identique à celle du champ ECI_Client_Series_Id définie dans le Tableau 7.4.3-1.

Tableau 7.4.4-1 – Définition de l'identificateur de client

Syntaxe	Nbre de bits	Mnémonique
`ECI_Client_Id {`		
`padding(4)`		
`type` `/* voir le Tableau 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Sémantique:

type: nombre entier	Valeur conforme au Tableau 5.2-2
client_type: nombre entier	Type de client attribué par l'**Autorité de confiance ECI**.
client_version_major: integer	Numéro de version majeure du **Client ECI** d'un **type de Client ECI**. Les versions augmentent de façon incrémentielle en cas de nouvelle version majeure.
client_version_minor: integer	Numéro de version mineure du **Client ECI**. La comparaison avec un numéro de version mineure dans les listes de révocation de **Clients ECI** peut entraîner la révocation de ces clients.

7.5 Formats de la Chaîne d'opérations de plate-forme ECI

7.5.1 Vue d'ensemble

La chaîne d'authentification du **Certificat d'opération de plate-forme** et les signatures des clients de l'**Opération de plate-forme** sont présentées dans la Figure 7.5.1-1. Elle commence par la **Liste de révocation des Opérateurs**, suivie du **Certificat d'Opérateur et** de la **Liste de révocation des opérations de plate-forme**. Elle finit par le **Certificat d'opération de plate-forme** contenant la clé publique d'**Opération de plate-forme**. Elle s'utilise en combinaison avec la **Liste de révocation** des clients de l'**Opération de plate-forme** afin de valider les **images de Clients ECI** autorisées à fonctionner sur la plate-forme concernée.

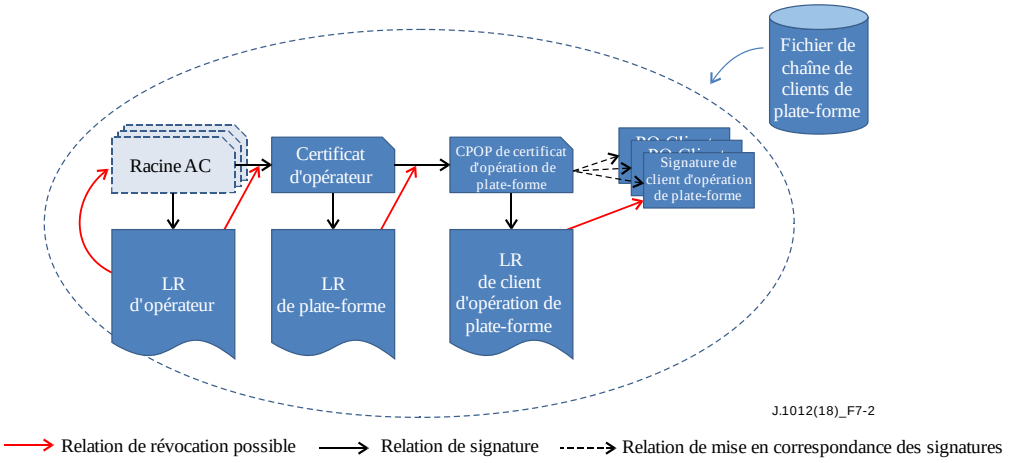


Figure 7.5.1-1 – Chaîne d'authentification de la chaîne de clients de la plate-forme

7.5.2 Certificats d'Opérateur

Les **Certificats d'Opérateur** sont définis par la structure **ECI_Certificate**. L'identificateur d'**Opérateur** est défini dans le Tableau 7.5.2-1.

Tableau 7.5.2-1 – Définition de l'identificateur d'Opérateur

Syntaxe	Nbre de bits	Mnémonique
ECI_Operator_Id {		
padding(4)		
type /* voir le Tableau 5.2-2	4	uimsbf
operator_id	20	uimsbf
operator_version	8	uimsbf
}		

Sémantique:

type: octet	Valeur conforme au Tableau 5.2-2.
operator_id: integer	**Identificateur d'Opérateur** attribué à un **Opérateur**, unique dans le contexte de la racine **ECI.**
operator_version: integer	Numéro de version attribué de manière incrémentielle à la version du **Certificat** de l'**Opérateur**. Les valeurs 0x00 et 0xF0..0xFF sont réservées.

7.5.3 Certificats d'opérations de plate-forme

Les **Certificats d'opérations de plate-forme** sont définis par la structure **ECI_Certificate**. La clé secrète de l'**Opération de plate-forme** est gérée par l'**Opération de plate-forme**. L'identificateur du **Certificat d'opération de plate-forme** est défini dans le Tableau 7.5.3-1.

Tableau 7.5.3-1 – Définition de l'identificateur d'Opération de plate-forme

Syntaxe	Nbre de bits	Mnémonique
ECI_Platform_Operation_Id {		
padding(4)		
type /* voir le Tableau 5.2-2	4	uimsbf
platform_operation_id	20	uimsbf
platform_operation_version	8	uimsbf
}		

Sémantique:

type: octet	Valeur conforme au Tableau 5.2-2.
platform_operation_id: integer	Numéro d'**Opération de plate-forme** attribué au Fournisseur de systèmes de sécurité, unique dans le contexte du **Certificat** de l'opérateur**.**
platform_operation_version: integer	Augmente au cas où l'**Opération de plate-forme** modifie son **Certificat**.

7.5.4 Liste de révocation des clients d'Opération de plate-forme

La liste de révocation des clients d'**Opération de plate-forme** est définie au § 5.3. Elle utilise l'attribution d'identificateur définie dans le Tableau 5.2-2. Les champs entity_id de la liste de révocation font référence au champ cosignature_id de la structure des données de signature des clients d'**Opération de plate-forme**.

Le numéro de version minimum de la liste de révocation est défini dans le cadre de l'initialisation du **Client ECI** et validé à l'aide du Système de sécurité avancée.

7.5.5 Cosignature de client d'Opération de plate-forme

La cosignature de client d'**Opération de plate-forme** fournit la signature d'**Opération de plate-forme** permettant de vérifier qu'une Image de client est autorisée à donner accès aux services

d'une plate-forme, ainsi que l'identificateur du fournisseur et du client de l'image, ce qui facilite la mise en correspondance avec l'image de client associée. Les signatures de Clients d'**opérations de plate-forme** possèdent leur propre énumération d'identificateur qui permet de révoquer de manière indépendante les **images de Clients ECI** précédemment autorisées par la liste de révocation des clients d'**opérations de plate-forme**. Le Tableau 7.5.5-1 fournit des détails complémentaires.

Tableau 7.5.5-1 – Définition de la cosignature des clients d'opérations de plate-forme

Syntaxe	Nbre de bits	Mnémonique
ECI_PO_Cosignature_Id {		
padding(4)		
type	4	uimsbf
entity_id	20	uimsbf
version	8	uimsbf
}		
ECI_PO_Client_Cosignature_Data {		
ECI_PO_Cosignature_Id **cosignature_id**	32	
client_tag	4	uimsbf
réservé	28	
ECI_Vendor_Id **vendor_id**	32	
if (/* image series cosignature */) {		
ECI_Client_Series_Id **client_series_id**	32	
format_version	8	uimsbf
if (format_version == 0x01){		
ECI_Signature_v1 **series_cosignature**		
}		
}		
if (/* image cosignature */){		
ECI_Client_id **client_id**	32	
ECI_Data_Signature **image_cosignature**		
}		
}		

Sémantique:

type: octet	Valeur conforme au Tableau 5.2-2.
entity_id: nombre entier	**Identificateur** unique attribué à la signature dans le contexte du **Certificat d'opération de plate-forme**. En conjonction avec le champ cosignature_version attribué à une seule **image de client autorisée**.
version: nombre entier	Augmenté (par exemple, en augmentant les bits de poids le plus fort) si l'**Opération de plate-forme** modifie sa clé publique. Les bits de moindre poids du champ peuvent servir à représenter (en partie) la version de la **Série d'images** de client ou l'image de client pour faciliter la gestion par l'**Opération de plate-forme** de la révocation en fonction de la version de client à l'aide du champ de version de la liste de révocation des clients d'**Opération de plate-forme**.
cosignature_id: ECI_PO_Cosignature_Id	Identification de l'identificateur de la cosignature sur une image de client. Ce champ est inclus dans le calcul de la cosignature.
client_tag: integer	Identificateur court utilisé aux fins d'installation pour désigner un type de client (client_type) dans le contexte d'une **Opération de plate-forme**. Seuls les clients interchangeables du point de vue de l'**Utilisateur** auront la même valeur client_tag. En général, les versions mineures d'un client sont équivalentes.
vendor_id: ECI_Vendor_Id	Identificateur du **Certificat** de fournisseur de l'**Image de client ECI**. Ce champ peut servir à localiser la **Série d'images** de client ou l'image de client dont cette structure de données fournit la cosignature.
client_series_id: ECI_Client_series_id	Identificateur du **Certificat** de série de clients servant à vérifier une image. Le champ de type du champ client_series_id correspondra au type enfant des **Certificats d'opération de plate-forme** de client_image_series (voir le Tableau 5.2-2). De ce fait, il définit la sélection correcte des autres interprétations possibles de la structure de données.
format_version	Version du format de la définition du **Certificat** s'appliquant à la cosignature (voir le Tableau 5.2-1). Elle correspondra à la définition de la version du **Certificat** de client. La seule valeur valable de ce champ est définie à 0x01.
series_cosignature: ECI_Signature_v1	Cosignature par la clé secrète d'**Opération de plate-forme** du certificat de série d'images de client. Les données entrant dans le calcul de la signature seront définies comme identiques à la valeur du champ de certificat client_image_series. Elles remplaceront l'identificateur de série d'images de client (client_image_series_id) par l'identificateur de cosignature (cosignature_id) de cette structure de données et le champ d'extension par une extension de 4 octets contenant le champ client_image_series_id d'origine du **Certificat**.
client_id: ECI_Client_Id	Identificateur de l'image de client. Le champ de type de client_id correspondra au type enfant des **Certificats d'opération de plate-forme** de l'image de client (client_image) (voir le tableau 5.2.2). De ce fait, il définit la sélection correcte des autres interprétations possibles de la structure de données.
image_cosignature: ECI_Data_Signature	Cosignature de l'image de client par la clé secrète d'**Opération de plate-forme**. Les données entrant dans le calcul de la signature seront définies comme suit: champ cosignature_id suivi par l'entrée des données du fichier d'image de client dans le calcul de la signature de l'image de client, comme défini au § 7.6.1.

7.6 Formats de fichiers

7.6.1 Format de fichier d'Image de client ECI

Le justificatif d'identité du **Client ECI** contient les données requises pour vérifier l'authenticité de l'**Autorité de confiance ECI** d'un **Client ECI**. Il utilisera le format défini dans le Tableau 7.6.1-1.

Tableau 7.6.1-1 – Définition des justificatifs d'identité des clients

Syntaxe	Nbre de bits	Mnémonique
ECI_Client_Credentials {		
ECI_certificate_Chain **client_chain**		
if (client_chain.chain_length == 0x1) { /* no client series; regular image */		
ECI_RL **client_rl**		
}		
ECI_Data_Signature **client_signature**		
}		

Sémantique:

header: ECI_Client_Chain_Header	En-tête de fichier de chaîne de **Clients ECI**.
client_chain: ECI_Client_Chain	**Chaîne de certificats** permettant de valider une **Image de client ECI.** Elle commence par la liste de révocation de racine des fournisseurs de systèmes de sécurité et se termine par le certificat de fournisseur de systèmes de sécurité pour les **Clients ECI** non basés sur une Série d'images ou par le **Certificat** de série de **Clients ECI** pour les **Clients ECI** basés sur une Série d'images.
client_rl: ECI_RL	Liste de révocation des identificateurs d'**Image de client ECI**.
client_signature: ECI_Data_Signature	Signature validant l'**Image de client ECI**, la clé publique fournie par la **Chaîne de clients ECI**.

Le fichier d'**Image de client ECI** est défini dans le Tableau 7.6.1-2.

Tableau 7.6.1-2 – Définition du fichier d'Image de client ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Client_Image_File {		
magic = 'ECI'	24	uimsbf
image_header_version	8	uimsbf
ECI_Client_Credentials **credentials**		
if (image_header_version == 0x01) {		
if (credentials.client_chain.chain_length == 0x1) { /* regular image */		
ECI_Client_Id **client_id**	32	uimsbf
}		
if (credentials.client_chain..chain_length == 0x2) { /* Image Series image*/		
ECI_ Image_Target_Id _Id **target _id**	64	uimsbf
ECI_Client_Series_Id **client_series_id**	32	
}		
vendor_id	20	uimsbf
image_encrypted_flag	14	uimsbf
online_flag	1	uimsbf
Réservé	10	
for (i=0; i<n; i++) {		
client_image_byte	8	uimsbf
}		
}		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il possède la valeur de trois représentations ASCII 8 bits des caractères 'ECI'. Afin de renforcer l'intégrité des données, l'**Hôte ECI** contrôlera la valeur de ce champ pour vérifier si un fichier **ECI** présente le format attendu.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. L'**Hôte ECI** ignorera les images dont il ne reconnaît pas le numéro de version.
credentials: ECI_Client_Credentials	**Justificatif d'identité de Client ECI** servant à vérifier l'authenticité de l'**Image de client ECI.**
series_image: booléen	series_image n'est pas un champ mais une fonction calculée à partir du justificatif d'identité indiquant la présence d'un **Certificat** de série de types de **Client ECI**.
series_id: ECI_Client_Series_Id	Identificateur de série de **Clients ECI** de la Série d'images de l'image suivante. L**'Hôte ECI** vérifiera la valeur avant de charger l'**Image de client ECI**.
series_image_id: ECI_Client-series_Image_Id	Identificateur d'image dans la Série d'images de l'image suivante. L**'Hôte ECI** vérifiera la valeur avant de charger l'**Image de client ECI**.
client_id: ECI_Client_Id	Identificateur de **Client ECI** de l'**Image de client ECI.** L**'Hôte ECI** vérifiera la valeur avant de charger l'**Image de client ECI**.
vendor_id: ECI_Vendor_Id	Identificateur du Fournisseur de systèmes de sécurité de l'**Image de client ECI** défini dans la structure ECI_Vendor_Id traitée dans le § 7.4.2. L'**Hôte ECI** contrôlera ce champ avant de charger une (nouvelle) **Image de client ECI**.
image_encrypted_flag: nombre entier	Ce drapeau indique si l'image est chiffrée. Si ce champ a pour valeur 0b0, l'image n'est pas chiffrée. S'il a pour valeur 0b1, elle est chiffrée.
online_flag: nombre entier	Ce drapeau indique si le protocole d'extraction d'une clé de déchiffrement de l'image requiert une interaction en ligne avec le serveur de préconfiguration à l'aide d'un nonce. Voir le § 7.8.3.
client_image_byte: octet	Séquence d'octets contenant l'image de client.

Dans le Tableau 7.6.1-2, la phrase "L'**Hôte ECI** contrôlera" signifie qu'il vérifiera que les valeurs qu'il attend correspondent à la valeur du champ.

La signature d'**Image de client ECI** sera calculée à partir de toutes les données du champ de justificatif d'identité suivant.

7.6.2 Données de la Chaîne d'opérations de plate-forme

Le fichier d'**Image de client ECI** est défini dans le Tableau 7.6.2-1.

Tableau 7.6.2-1 – Définition du fichier de la Chaîne d'opérations de plate-forme

Syntaxe	Nbre de bits	Mnémonique
ECI_Operation_certificate_File {		
magic = 'EPC'	24	uimsbf
version	8	uimsbf
if (version == 0x01) {		
ECI_certificate_Chain **operation_chain**		
ECI_RL po_client_rl		
client_image_count	16	uimsbf
for (i=0; i<client_image_count; i++) {		
ECI_PO_Client_Cosignature_Data **po_client_data**		
}		
ECI_RL po_client_rl		
}		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il possède la valeur de trois représentations ASCII 8 bits des caractères 'EPC'. Afin de renforcer l'intégrité des données, l'**Hôte ECI** contrôlera la valeur de ce champ pour vérifier si un fichier **ECI** présente le format attendu.
Image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. L'**Hôte ECI** ignorera les images dont il ne reconnaît pas le numéro de version.
operation_chain: ECI_Client_Chain	**Chaîne de certificats** servant à valider une **Image de client ECI**. Elle commence par la liste de révocation de la Racine de l'opérateur et se termine par le **Certificat d'opération de plate-forme.**
po_client_rl: ECI_RL	Liste de révocation des clients d'**Opération de plate-forme** servant à valider les cosignatures d'images de client. L'**Hôte ECI** vérifiera les identificateurs de cosignature (cosignature_ids) du champ po_client_data dans le cadre de la vérification de la cosignature.
client_image_count: integer	Nombre de structures des données de signature des images de client dans la boucle suivante.

Dans le Tableau 7.6.2-1, la phrase "L'**Hôte ECI** contrôlera" signifie qu'il vérifiera que les valeurs qu'il attend correspondent à la valeur du champ.

7.6.3 Fichiers de données de révocation

Pour le **chargeur de Client ECI** il existe deux types de fichiers de données de révocation utilisant tous les deux le format ECI_Revocation_Data_File défini dans le Tableau 5.5-2.

Le fichier de données de révocation des **Clients ECI** utilise un champ father_type égal à 0x0 (**Certificat racine**) et un champ sub_type égal au type de liste de révocation des Fournisseurs. Le champ revocation_data respecte la contrainte imposant que la liste de révocation feuille des arborescences corresponde à des listes de révocation de **Clients ECI**.

Le fichier de données de révocation de l'**Opération de plate-forme** utilise un champ father_type égal à 0x0 (**Certificat racine**) et un champ sub_type égal au type de liste de révocation de l'**Opérateur**. Le champ revocation_data respecte la contrainte imposant que les listes de révocation feuilles des arborescences correspondent à des listes de révocation de l'**Opération de plate-forme**.

7.7 Protocoles de transport des ressources des Clients ECI

7.7.1 Généralités et profil

Ce paragraphe définit l'application des protocoles dans les **Équipements CPE** et les **opérations de plate-forme**.

Le protocole de radiodiffusion ne propose pas l'option **Série d'images**. Les images basées sur des séries ne sont prévues que sur les dispositifs connectés sur IP.

Les **Équipements CPE** prenant en charge à la fois l'accès en mode radiodiffusion et en ligne aux ressources des **Clients ECI** utiliseront prioritairement l'accès en mode radiodiffusion (sauf indication contraire dans la présente Recommandation) afin de délester le trafic en ligne, mais pourront utiliser l'accès en ligne en cas d'urgence (attente de l'**Utilisateur**) et s'il n'est pas possible d'obtenir les fréquences d'accès minimales sur le réseau de radiodiffusion.

7.7.2 Protocole de transport en mode radiodiffusion

7.7.2.1 Introduction

Afin d'initialiser et de prendre en charge le **Client ECI,** l'interface **ECI** nécessite des données de prise en charge de diverses fonctions au nom du **Client ECI** et/ou de l'**Hôte ECI**. Tous les types de données utilisent le même protocole de transport, défini dans le présent paragraphe et très proche du protocole de téléchargement des fichiers d'**Images de l'hôte ECI**.

Dans le cas de la livraison en mode radiodiffusion, les données sont fractionnées en seaux à l'aide d'une fonction de hachage appliquée à l'indice d'accès qu'utilise l'**Équipement CPE** pour déterminer

s'il a besoin de ces données. Le recours à des seaux réduit de façon importante la quantité de données que l'**Équipement CPE** doit télécharger et améliore la sélectivité du suivi des changements apportés aux données pertinentes pour l'**Équipement CPE**.

Les groupes de carrousels distincts suivants sont définis (par type de contenu):

- Images de client ECI (par Fournisseur de systèmes de sécurité).
- Données de révocation de **Client ECI**, structurées en seaux sur la base du champ <client_id, client-version_major> et de l'indice d'identificateur de fournisseur (vendor_id).
- **Chaîne de certificats** d'opération de plate-forme.
- Données de révocation d'**Opération de plate-forme**, structurées en seaux sur la base de l'identificateur de fournisseur (provider_id) et de l'indice d'identificateur d'opérateur (operator_id).
- Données de révocation d'**Image de l'Hôte ECI**, structurées en seaux.
- Configuration du système de sécurité évoluée **ECI** (AS_setup), données d'initialisation du **Client ECI** structurées en seaux.
- Des groupes de carrousel sont définis pour les structures de données destinées à l'importation et à l'exportation (voir le § 9.8).
- Les groupes de carrousels sont définis dans les données propriétaires de l'**Opérateur**.

Tous les paramètres des carrousels DSMCC seront conformes à [ETSI EN 301 192].

Un **Opérateur** pourra utiliser plusieurs carrousels sur différents multiplexes pour transmettre l'ensemble des données requises. Cependant, pour un **Client ECI** spécifique, l'**Hôte ECI** n'aura à surveiller que les mises à jour d'un unique message de l'infrastructure d'identité numérique (DII) d'emplacement d'un carrousel de données.

7.7.2.2 Transmission à l'Opérateur des justificatifs d'identité et des données de révocation

Les formats de données et les protocoles de transfert des justificatifs d'identité et des listes de révocation à un **Opérateur** ne font pas partie de la spécification de l'interface **ECI**.

7.7.2.3 Transfert du Fournisseur de systèmes de sécurité à l'Opérateur

Les formats de données et les protocoles de transfert de contenus du **Fournisseur de systèmes de sécurité** à l'**Opérateur** ne font pas partie de la présente Recommandation.

7.7.2.4 Signalisation PSI

Les carrousels utiliseront le descripteur stream_identifier_descriptor [ETSI EN 300 468] dans la table de correspondance du programme (PMT) pour baliser le flux de transmission du carrousel afin d'en permettre le référencement par le descripteur data_broadcast dans le système SI.

Les carrousels utiliseront un descripteur data_broadcast_id_descriptor avec l'identificateur data_broadcast_id défini dans le Tableau 7.7.2.4-1.

Tableau 7.7.2.4-1 – Valeur de l'identificateur de radiodiffusion des données pour des carrousels ECI particuliers

Valeur de Data_broadcast_id	Signification
Allouée par le bureau des projets DVB; voir la valeur de l'identificateur de radiodiffusion définie dans [ETSI TS 101 162].	Carrousel de données de prise en charge d'un client particulier par l'**Opérateur ECI**

Les octets de sélection du descripteur data_broadcast_id_descriptor seront conformes à la structure définie dans le Tableau 7.7.2.4-2.

Tableau 7.7.2.4-2 – Structure de l'identificateur des carrousels de données ECI DVB DSMCC

Syntaxe	Nbre de bits	Mnémonique
ECI_carousel_id_structure {		
version	8	uimsbf
if (version == 0x01){		
operator_id	20	uimsbf
platform_**operation_id**	20	uimsbf
}		
}		

Sémantique:

version: nombre entier	Version de la structure. Actuellement uniquement définie à 0x01. Toutes les autres valeurs sont réservées. Les **Équipements CPE** rencontrant une autre version ignoreront ce descripteur.
operator_id: ECI_Operator_Id	Identificateur **ECI** de l'**Opérateur** (défini pour n'importe quel **Certificat d'Opérateur**) de l'**Opération de plate-forme** du carrousel.
platform_operation_id: ECI_Platform_Operation-Id	Conforme au **Certificat d'opération de plate-forme:** Identificateur de l'**Opération de plate-forme**.

7.7.2.5 Signalisation SI

7.7.2.5.1 Signalisation de l'emplacement du carrousel de données via le descripteur de lien d'emplacement de données

Le descripteur de lien d'emplacement de données de **Client ECI** est un descripteur de lien DVB privé **ECI** [ETSI TS 101 162]. Il aide l'**Équipement CPE** à localiser le multiplexe contenant un carrousel de données de **Client ECI** pour une **Opération de plate-forme** particulière. Il se trouve dans la table NIT ou BAT. Le descripteur de lien d'emplacement de données de **Client ECI** sera toujours précédé dans la section de la table par un descripteur spécificateur de données privées DVB [ETSI TS 101 162] dont la valeur du champ private_data_specifier est égale à "ECI", comme défini dans [ETSI TS 101 162]. Il pourra apparaître plusieurs fois dans la table NIT ou BAT. Il se trouvera dans les réseaux et les bouquets comportant plus de quatre multiplexes.

Par rapport à la définition du descripteur de lien figurant dans [ETSI EN 300 468] et [ETSI TS 101 211], les champs du descripteur de lien d'emplacement de données de **Client ECI** présentent l'application particulière suivante:

- **service_id:** peut être mis à 0x0000 pour indiquer qu'aucun identificateur de service (service_id) particulier n'est signalé.
- **linkage_type:** valeur égale à 0x80 signalant un descripteur de lien d'emplacement de données de client ECI.

Le champ d'octets de données privées du descripteur de lien d'emplacement de données de **Client ECI** présentera la structure définie dans le Tableau 7.7.2.5.1-1.

Tableau 7.7.2.5.1-1 – Structure des données privées du descripteur de lien d'emplacement du carrousel de données d'un Client ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_client_data_location {		
version	8	uimsbf
if (version==0x01){		
for (i=0; i<n; i++) {		
operator_id	20	uimsbf
platform_**operation_id**	20	uimsbf
}		
}		
}		

Sémantique:

version: nombre entier	Version de la structure. Actuellement uniquement définie à 0x01. Toutes les autres valeurs sont réservées. Les **Équipements CPE** rencontrant une autre version ignoreront ce descripteur.
operator_id: ECI_Operator_Id	Identificateur **ECI** de l'**Opérateur** (défini pour n'importe quel **Certificat d'Opérateur**) de l'**Opération de plate-forme** du carrousel. La valeur 0x00000 signale n'importe quel **Opérateur**.
platform_operation_id: ECI_Platform_Operation-Id	Conforme au **Certificat d'opération de plate-forme:** Identificateur de l'**Opération de plate-forme**. La valeur 0x0000 signale n'importe quelle **Opération de plate-forme**.

Les opérateurs de réseaux et de bouquets peuvent utiliser des spécificateurs génériques (valeur 0x00000) pour l'identificateur d'opérateur (operator_id) ou d'opération de plate-forme (platform_operation_id) afin d'établir la liaison avec un multiplexe contenant un ou plusieurs carrousels de données de **Client ECI**. Pour des raisons d'efficacité, il est recommandé de contraindre ce type de signalisation à aider les **Équipements CPE** à inspecter le nombre minimal de multiplexes requis pour localiser un carrousel d'**Opération de plate-forme** particulier.

Il est conseillé de n'utiliser dans une table NIT ou BAT qu'un seul descripteur de lien d'emplacement de carrousel de données de **Client ECI** avec un multiplexe et d'indiquer tous les carrousels concernés situés dans ce multiplexe selon une unique structure ECI_Client_data_location.

7.7.2.5.2 Descripteur de téléchargement d'urgence de Client ECI

Afin d'indiquer la nécessité de remplacer d'urgence une **Image de client ECI**, un ou plusieurs descripteurs ECI_client_emergency_download pourront être placés dans les tables NIT et BAT ou dans l'une des entrées de la table de description des services (SDT) pour les services auxquels le **Client ECI** concerné peut fournir l'accès. L'**Hôte ECI** pourra extraire ce descripteur de l'une quelconque des tables dans lesquelles il apparaît dans les multiplexes en cours de syntonisation, exécuter le traitement associé et utiliser n'importe quel syntoniseur de rechange pour accéder aux multiplexes où il pourra acquérir ce descripteur en 30 minutes dans le pire des cas.

Le champ ECI_client_emergency_download_descriptor permet de cibler certaines plates-formes d'opérations ainsi que des types d'hôte particuliers afin de réduire la gêne engendrée par les mises à jour d'urgence.

Lorsque l'**Hôte ECI** trouve un nouveau descripteur ECI_client_emergency_download (vérifié en fonction de l'origine de la table et du champ emergency_id), il compare sa configuration Hôte/Client aux informations de ciblage figurant dans le descripteur. S'il trouve une cible correspondante et si la version de l'image de client déjà installée requiert une mise à jour, l'Hôte l'effectuera conformément à la valeur du champ emergency_indicator. En cas de conflit de ressources, cette action est susceptible de perturber les activités en cours des **Utilisateurs** de l'**Équipement CPE**.

Le descripteur d'opération **ECI** est un descripteur DVB privé. Dans la table où il apparaît, il sera toujours précédé du champ DVB private_data_specifier_descriptor utilisant le champ **ECI** private_data_specifier_field (voir [ETSI EN 300 468]). La syntaxe du descripteur est définie dans le Tableau 7.7.2.5.2-1.

Tableau 7.7.2.5.2-1 – ECI_Client_Emergency_Download_Descriptor

Syntaxe	Nbre de bits	Mnémonique
ECI_client_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* main loop */		
main_loop_nr	8	uimsbf
for (i=0; i<main_loop_nr; i++){		
/* target platform */		
platform_operation_tag	8	uimsbf
/* host target loop */		
host_nr	8	uimsbf
/* host id target loop */		
for (j=0; j<host_nr; j++){		
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
host_version	8	uimsbf
}		
/* client image loop */		
client_nr		
for (j=0; j<client_nr; j++){		
emergency_indicator	4	uimsbf
client_tag	4	uimsbf
min_client_version_major	8	uimsbf
min_client_version_minor	8	uimsbf
}		
}		
/* private data till end of descriptor*/		
for (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Sémantique:

descriptor_tag	Valeur d'étiquette privée **ECI** du champ descriptor_tag: voir le document [b-UIT-T J Suppl. 7].
descriptor_length	Voir [b-ETSI EN 300 468].
main_loop_nr	Nombre d'entrées dans la boucle principale. Les entrées de boucle principale seront évaluées individuellement par l'**Hôte ECI**; en d'autres termes, elles auront une sémantique OR. Les divers éléments d'une entrée de boucle auront une sémantique AND.
platform_operation_tag	Valeur d'étiquette de plate-forme **ECI** figurant dans le champ ECI_platform_operation_descriptor des tables NIT/BAT. L'**Hôte ECI** envisagera une mise à jour d'urgence si le champ platform_operation correspond au champ platform_operation de l'un des **Clients ECI** installés.
host_nr	Nombre d'entrées dans la boucle d'hôte cible. La valeur 0 signifie que tous les **Hôtes ECI** sont ciblés. Les entrées de boucle auront une sémantique OR si l'une quelconque des spécifications de l'hôte correspond à la condition de cible dans la boucle principale et présente l'état correspondant.
manufacturer_id	Identificateur de Fabricant de l'hôte cible d'une mise à jour d'urgence. L'hôte envisagera une mise à jour d'urgence si la valeur de ce champ correspond à l'identificateur de Fabricant (manufacturer_id) de l'**Hôte ECI**.

cpe_type_id	Valeur définie par le champ ECI_CPE_Type_ID dans le Tableau 6.2.2.1-2. L'**Hôte ECI** envisagera une mise à jour d'urgence si la valeur du champ cpe_type_id de l'Hôte correspond à celle de ce champ. Une valeur 0x000 pour le champ cpe_type_id.cpe_type signifiera que le champ cpe_type de l'**Hôte ECI** correspond (et les champs cpe_model et host_version seront ignorés). Une valeur 0x00 du champ cpe_type_id.cpe_model signifiera que n'importe quel modèle d'équipement CPE (cpe_model) d'**Hôte ECI** correspond (et la version de l'Hôte sera ignorée).
host_version	L'**Hôte ECI** envisagera une mise à jour d'urgence si, et seulement si, sa version est inférieure ou égale à la valeur de ce champ. Voir la Note.
client_nr	Nombre d'entrées dans la boucle d'images de client. Les entrées de boucle possèderont une sémantique OR et toutes les images de client correspondantes seront examinées afin de déterminer l'opportunité d'une mise à jour d'urgence.
emergency_indicator	L'**Hôte ECI** utilisera la valeur de ce champ pour sélectionner le comportement permettant de démarrer le téléchargement puis de mettre à jour le client, comme défini dans le Tableau 7.7.2.5.2-2.
client_tag	Valeur d'étiquette identifiant le **Client ECI** figurant dans le champ ECI_platform_operation_descriptor des tables NIT/BAT correspondant au champ platform_operation_tag dans la même boucle principale. L'**Hôte ECI** envisagera une mise à jour d'urgence si les champs vendor_id et client_id correspondent à l'un des clients installés sur l'**Hôte ECI**.
min_client_version_major	Ce champ indique le numéro de version majeure minimal acceptable pour l'image de client. L'**Hôte ECI** envisage d'effectuer une mise à jour d'urgence si la version majeure d'un Client installé correspondant au champ client_tag est inférieure à la valeur de ce champ.
min_client_version_minor	Ce champ représente le numéro de version mineure minimal acceptable pour l'image de client. L'**Hôte ECI** envisagera d'effectuer une mise à jour d'urgence si la version mineure d'un **Client ECI** installé correspondant au champ client_tag est inférieure à la valeur de ce champ et si sa version majeure est égale à la valeur du champ min_client_version_major.
client_id	Identificateur d'un **Client ECI** fournissant des services de déchiffrement pour des services associés au champ platform_operation_tag, comme défini dans le Tableau 7.4.4-1.
private_data_byte	Données privées: les contenus peuvent être définis par l'**Opérateur** qui gère la diffusion de ce descripteur.
NOTE – Une valeur de champ égale à 0xFF implique que toutes les versions de l'Hôte correspondent.	

Le Tableau 7.7.2.5.2-1 définit plusieurs conditions de la boucle principale (dotée d'une sémantique AND) à satisfaire pour que l'**Hôte ECI** envisage l'exécution d'une mise à jour d'urgence. Si toutes ces conditions sont réunies, l'**Hôte ECI** procédera au téléchargement et à l'installation d'urgence d'une ou plusieurs images de client conformément à la valeur du champ emergency_indicator du client concerné.

Tableau 7.7.2.5.2-2 – Valeurs des champs ECI_Client_emergency_download_descriptor et emergency_indicator

Nom	Valeur	Description
System emergency	0x01	L'**Hôte ECI** téléchargera la nouvelle image de client, l'installera le plus rapidement possible et interrompra les activités en cours initiées par l'**Utilisateur** si nécessaire (voir la NOTE 1).
Client emergency	0x02	L'**Hôte ECI** téléchargera la nouvelle image de client et l'installera avant l'ouverture d'une quelconque session de pointeur de média pour ce Client. Il sera d'abord mis fin aux sessions de pointeur de média en cours pour le Client en question (voir la NOTE 2).
Client urgency	0x03	L'**Hôte ECI** téléchargera la nouvelle image de client et l'installera dès que possible sans perturber les activités initiées par l'**Utilisateur.** L'**Hôte ECI** téléchargera la nouvelle image de client au plus tard lors de la prochaine mise sous tension (voir la NOTE 3).
RFU	autre	Réservé à une utilisation future.
NOTE 1 – Les **Opérateurs** peuvent utiliser cette option, par exemple si le **Client ECI** actuel est susceptible de présenter un risque pour l'**Hôte ECI** et/ou d'autres **Clients ECI** et doit être remplacé immédiatement. NOTE 2 – Les **Opérateurs** peuvent utiliser cette option, par exemple en cas de mauvaises performances de déchiffrement du **Client ECI** actuel. NOTE 3 – Les **Opérateurs** peuvent utiliser cette option, par exemple si le **Client ECI** actuel présente de graves défaillances en matière de services de déchiffrement mais peut fonctionner sans difficulté majeure dans les cas d'utilisation normale.		

7.7.2.6 Descripteur de compatibilité des carrousels

Le descripteur de compatibilité utilisé dans les carrousels de données DVB DSMCC [ETSI EN 301 192] servira également dans les messages DSI DII.

Il fournit des informations sur le type de données transportées dans un groupe de carrousels. Le champ specifierData() contiendra l'identificateur unique d'organisation **ECI**. Le Tableau 7.7.2.6-1 définit les champs applicables du descripteur de compatibilité dans les carrousels de données de **Clients ECI**.

Tableau 7.7.2.6-1 – Types de contenu des carrousels de données ECI

Champ de type de descripteur	Objet du groupe	Champ de modèle	Champ de version	Indice de seau aux fins de calcul de l'identificateur du module
0xA0	**images de Client ECI** et fichiers de justificatifs d'identité d'un Fournisseur	Identificateur du **Fournisseur de systèmes de sécurité** des images.		Attribué librement
0xA2	Fichiers de données de révocation de **Client ECI** (sous forme de seaux)	platform_operation_id		= Vendor_id + <Client_type, client_version_major> (voir la NOTE)
0xA3	Fichier de chaîne d'**opérations de plate-forme**	platform_operation_id, platform_operation_version		Attribué librement
0xA4	Fichiers de données de révocation d'**Opération de plate-forme** (sous forme de seaux)	platform_operation_id		= Operator_id + provider_id
0xA5	Fichiers de données de révocation d'**Hôte ECI** (sous forme de seaux)	platform_operation_id		= Manufacturer_id + cpe_type_id
0xA6	Fichiers de configuration de système de sécurité évoluée (AS_setup) (sous forme de seaux)	platform_operation_id		Identificateur de cible (target_id) de l'**Équipement CPE**
0xA7-0xAA	Conteneur d'applications d'interface d'utilisateur (voir le § 9.4.3.4.2)	Défini par l'**Opérateur**		Attribué librement
0xB0	Fichier d'arborescence d'exportation	Identificateur d'opération de plate-forme (platform_operation_id) (du **Client ECI** exportateur)		Attribué librement
0xB1	Fichier de chaînes d'importation	Identificateur d'opération de plate-forme (platform_operation_id) (du **Client ECI** importateur)		Attribué librement
0xB2	Fichier de chaîne d'authentification de l'importation	Identificateur d'opération de plate-forme (platform_operation_id) (du **Client ECI** importateur)		Attribué librement
0xB8-0xBF	Format propriétaire de l'opérateur	Défini par l'**Opérateur**		Défini par l'**Opérateur**
Autres valeurs	réservé			
NOTE – Concaténation des deux champs, dont le plus significatif constitue le premier argument, générant un nombre de 20 bits.				

Le calcul de l'indice des seaux utilisera une arithmétique à nombre entier modulaire de 32 bits. Il est défini au § 7.7.2.7.

7.7.2.7 Lancement du serveur de téléchargement des carrousels (DSI)

Si un carrousel comporte deux couches, le DSI contiendra un index complet des groupes présents dans le carrousel (à savoir une entrée de boucle par indication d'information relative au téléchargement [DII]).

Le descripteur de compatibilité est défini dans le Tableau 7.7.2.6-1. Les champs DII sans boucle respecteront les contraintes suivantes:

- block Size: 512 octets minimum. Pour les groupes comportant des modules importants, la taille minimum conseillée est de 2 kilooctets;
- tCDownloadScenario: au moins quatre fois le message de répétition de bloc de données à télécharger (DDB) le plus lent du groupe. tCDownload satisfera également aux contraintes maximales énoncées dans le Tableau B.4-1;

- numberOfModules: nombre de modules des carrousels normaux et nombre de seaux (chacun associé à un module) des données en seaux. Pour les données des **Chaînes de Certificats d'opération de plate-forme**, la valeur sera égale à 1.

Les valeurs ci-dessous du champ tCDownloadScenario correspondent à la période d'expiration de l'acquisition d'un élément de données complet par un **Équipement CPE**. Celle-ci sera au moins égale à quatre fois le temps de répétition DDB le plus lent de l'un des modules du groupe. Les valeurs des différents éléments sont définies au § B.4.

Les champs suivants des boucles de modules respecteront les contraintes ci-après:

- moduleId: les bits 15 à 8 seront identiques aux bits de plus faible poids du groupId dans la structure groupInfo correspondante du DSI. L'attribution des bits 7 à 0 est conforme au Tableau 7.7.2.7-1.
- moduleVersion: l'application dépend du type de carrousel. Elle sera conforme au Tableau 7.7.2.7-1.
- moduleInfoLength: 0 pour tous les carrousels **ECI**.

Tableau 7.7.2.7-1 – Paramètres des groupes de carrousels ECI

Type de groupe	ModuleId bit 7..0	ModuleVersion	ModuleInfo
images de client	client_type	client_version	Aucun
Données de révocation du client	bucket_number	Augmenté à chaque mise à jour	Aucun
Chaîne de clients d'**Opération de plate-forme**	Attribué par l'**Opérateur**	Augmenté à chaque mise à jour	Aucun
Données de révocation d'**Opération de plate-forme**	bucket_number	Augmenté à chaque mise à jour	Aucun
Données de révocation d'**Hôte ECI**	bucket_number	Augmenté à chaque mise à jour	Aucun
Données de configuration de sécurité évoluée ECI	bucket_number	Augmenté à chaque mise à jour	Aucun

Le numéro de seau des données en seau (égal aux bits [7...0] de module_id) sera calculé à partir de l'indice avec une simple opération modulo:

```
bucket_number = indice de seau % nombre de modules
```

7.7.2.8 Blocs de données à télécharger (DDB) des carrousels

Aucune exigence particulière.

7.7.2.9 Comportement dynamique des carrousels

La numérotation de la version des carrousels et la mise à jour du lancement du serveur de téléchargement et de l'indication d'information de téléchargement seront conformes à [ETSI TR 101 202]. Cela signifie que chaque mise à jour d'un module apparaîtra dans le numéro de version du module et son indication d'information de téléchargement et remontera en cascade jusqu'au lancement du serveur de téléchargement (s'il est présent).

La mise en œuvre des **Équipements CPE** peut surveiller les changements apportés à leurs modules cibles afin de suivre les éventuelles mises à jour dynamiques pendant le fonctionnement normal.

7.7.3 Protocoles de transport web

7.7.3.1 Introduction

L'**Hôte ECI** peut extraire les divers éléments de données requis d'un serveur à désigner par l'**Opérateur**.

L'interface utilisera les requêtes HTTPS directes comme indiqué au § 9.4.4.6 et suivra les principes de conception RESTfull [b-Richardson]. Le codage de la requête combinera l'extension de l'adresse URL et les paramètres de recherche; la réponse sera codée sous forme de fichier binaire.

Le serveur HTTP répondra par l'un des codes de statut suivants:

- 200: OK (le fichier demandé est fourni).
- 302 FOUND: redirige la requête vers un autre serveur; la requête HTTP est répétée sur l'adresse URL renvoyée.
- 404: l'élément est absent du serveur.
- 500 .. 599: erreur serveur.

La spécification des adresses URL utilisées pour les requêtes est du type 'Bachus Naur'. Le nom des symboles correspondants aux champs dans les structures de données de l'interface **ECI** sera représenté sous la forme hexadécimale (chaîne de caractères '0' .. '9', 'A' .. 'F') de sa valeur avec un nombre d'octets double du nombre de chiffres utilisé pour représenter le nombre dans les structures de données binaires internes de l'interface **ECI**. Le serveur ne tiendra pas compte des paramètres de recherche supplémentaires qu'il ne reconnaît pas.

7.7.3.2 Vue d'ensemble de l'API web de l'interface ECI

L'**Opérateur** prendra en charge un serveur en ligne répondant à la requête Get HTTP1.1 [IETF RFC 7231] suivante dont l'adresse URL respecte la syntaxe et la sémantique ci-après:

URL ::= base-url '/' 'eci' major '_' minor '/' tail.

major et **minor** correspondront aux numéros majeur et mineur de la version du protocole dans une représentation décimale sans zéro de tête. La version actuelle est 1.0. La définition de 'tail' est fournie dans le Tableau 7.7.3.2-1.

Tableau 7.7.3.2-1 – Définition de 'tail'

```
tail ::= host_version |
   host_images |
   host_image_version |
   host_image |
   po_check |
   po_client_check   po_certchain |
   po_revocation |
   client_version |
   client_credential_version |
   client_image |
   client_revocation |
   as_request |
   tail_extension*.
```

tail_extension indique les diverses options d'extensions de l'API web de l'interface **ECI** définies dans la présente Recommandation.

7.7.3.3 Requêtes relatives à l'Hôte ECI de l'API web

Les requêtes ci-après de l'API web en rapport avec l'**Hôte ECI** sont définies comme suit:

- host_version ::= 'host-version ' '?target-id=' target_id.
  Cette requête fournira la version la plus récente de l'**Image de l'Hôte ECI** définie pour l'**Équipement CPE** identifié par **target_id**.
- host_images ::= 'hi-images' ' ?target-id=' target_id.
  Cette requête fournira le nombre le plus récent d'images d'un **Hôte ECI** pour l'**Équipement CPE** identifié par **target_id**.
- host_image_version ::= 'hi-version' ' ?target-id=' target_id '&image-id=' image_id .
  Cette requête fournira la version la plus récente de l'identificateur d'image (**image_id**) du fichier d'**Image de l'Hôte ECI** pour l'**Équipement CPE** identifié par **target_id**.

- host_image ::= 'host-image' ' ?target-id=' target_id '&image-id=' image_id. Cette requête fournira le numéro d'image (**image_number**) de l'**Image de l'Hôte ECI** pour l'**Équipement CPE** identifié par **target_id**. image_number=="FF" renverra le fichier de justificatif d'identité de l'**Hôte ECI** pour les **Images de l'hôte ECI**, y compris les données de révocation les plus récentes.

Dans le cas de requêtes en rapport avec l'**Hôte ECI**, le serveur d'une **Opération de plate-forme** peut prendre en charge les **Hôtes ECI** du type d'**Équipement CPE** de son choix. S'il prend en charge un type d'**Équipement CPE**, il prendra en charge l'ensemble complet le plus récent des **Images de l'hôte ECI** ainsi que les demandes **host_image_version**, **host_images** et **host_revocation** correspondantes. Le fichier est renvoyé au format ECI_Host_Version_File défini dans le Tableau 7.7.3.3-1.

Tableau 7.7.3.3-1– Définition du fichier de version de l'Hôte ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_Version_File {		
magic = 'RHVE'	32	uimsbf
host_version	8	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RHIM'.
host_version: nombre entier	Numéro de version du **Certificat de l'Hôte ECI.**

Le fichier est renvoyé au format ECI_Host_images_File défini dans le Tableau 7.7.3.3-2.

Tableau 7.7.3.3-2 – Définition du fichier d'Images de l'hôte

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_images_File {		
magic = 'RHIM'	32	uimsbf
host_images	8	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RHIM'.
host_images: nombre entier	Nombre d'**Images de l'hôte ECI** prises en charge par le type d'**Équipement CPE** identifié dans la requête.

Le fichier est renvoyé au format ECI_Host__Image_Version_File défini dans le Tableau 7.7.3.3-3.

Tableau 7.7.3.3-3 – Syntaxe du fichier de version d'Image de l'hôte

Syntaxe	Nbre de bits	Mnémonique
ECI_Host_Image_Version_File {		
magic = 'RHIV'	32	uimsbf
host_image_version	16	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RHIV'.
host_image_version: nombre entier	Version de l'**Image de l'Hôte ECI** identifiée par la requête.

7.7.3.4 Requêtes relatives aux opérations de plate-forme de l'API web

Le serveur de l'**Opération de plate-forme** prendra en charge les requêtes suivantes pour le compte des identificateurs d'**Opération de plate-forme** qu'il prend en charge:

```
po_check ::= 'po_check' '/' operator_id '/' platform_operation_id .
```

Cette requête renverra le statut de révocation du **Certificat** émis pour **operator_id** et **platform_operation_id** au format de fichier défini dans le Tableau 7.7.3.4-1. Le serveur d'une **Opération de plate-forme** prendra au minimum en charge ses propres **Certificats d'opération de plate-forme** à l'aide de cette interface.

```
po_client_check ::= 'po-client-check' '/' operator_id '/'
    platform_operation_id '?cosignature-id=' cosignature_id .
```

Cette requête renverra le statut de révocation de la plate-forme de l'**Image de client ECI** pour cosignature_id conformément à la liste de révocation la plus récente des clients de l'opération de plate-forme. Voir le Tableau 7.7.3.4-2.

```
po_certchain ::= 'po-chain' '/' operator_id '/' platform_operation_id .
```

Cette requête renverra la chaîne de **Clients ECI** la plus récente pour l'**Opération de plate-forme** identifiée par **operator_id** et **platform_operation_id** définie dans le Tableau 7.6.2-1. Le serveur d'une **Opération de plate-forme** prendra au minimum en charge ses propres **Certificats d'opération de plate-forme** à l'aide de cette interface.

```
po_revocation_ ::= 'po-revoc' '/' operator_id .
```

Cette requête renverra le fichier de données de révocation de l'**Opération de plate-forme** le plus récent contenant la liste de révocation de l'**Opérateur** identifié par **operator_id**. Le serveur prendra au minimum en charge les données de révocation les plus récentes de l'**Opérateur** de sa propre **Opération de plate-forme**. Les **Hôtes ECI** utiliseront cette API pour s'efforcer d'acquérir les données de révocation les plus récentes relatives à l'ensemble des **Clients ECI** stockés.

Tableau 7.7.3.4-1 – Syntaxe du fichier de contrôle des opérations de plate-forme

Syntaxe	Nbre de bits	Mnémonique
ECI_PO_Check_File {		
magic = 'RPCH'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RHIV'.
non_revoked_Certificate_flag: octet	Valeur égale à 0x00 si le **Certificat** de l'identificateur d'**Opération de plate-forme** identifié par la requête a été révoqué. Elle est égale à 0x01 dans tous les autres cas.

Tableau 7.7.3.4-2 – Syntaxe du fichier de contrôle des Clients d'opérations de plate-forme

Syntaxe	Nbre de bits	Mnémonique
ECI_PO_Client_Check_File {		
magic = 'RPCC'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RHIV'.
non_revoked_Certificate_flag: octet	La valeur est égale à 0x00 si l'image de client associée au champ cosignature_id de la requête a été révoquée conformément à la liste de révocation des clients de l'opération de plate-forme concernée. Elle est égale à 0x01 dans tous les autres cas.

7.7.3.5 Requêtes des clients de l'API web

Le serveur de l'**Opérateur** prendra en charge les requêtes suivantes pour le compte des clients requis par son identificateur d'**Opération de plate-forme**:

```
client_version  ::=  'client-ver'  '/'  vendor_id  '/'
     client_type '/' client_version_major .
```

- Cette requête renverra un fichier de version de client (voir le Tableau 7.7.3.5-1) contenant la version d'**Image de client ECI** la plus récente d'un client identifié par **vendor_id** et **client_type**. Au minimum, le serveur prendra en charge les clients servant à fournir ses propres services d'**Opération de plate-forme**.

```
client_credential_version  ::=  'client-ver'  '/'  vendor_id  '/'
          client_type '/' client_version_major .
```

- Cette requête renverra un fichier de version de justificatif d'identité de client (voir le Tableau 7.7.3.5-2) contenant la version la plus récente du justificatif d'identité du **Client ECI** identifié par **vendor_id** et **client_type**. Au minimum, le serveur prendra en charge les clients servant à fournir ses propres services d'**Opération de plate-forme**.

```
client_image  ::=  'client-img'  '/'  vendor_id  '/'
          client_type  '/'  client_version_major
          ['? &target-id=' image_target_id] .
```

- Cette requête renverra le fichier d'**Image de client ECI** le plus récent d'un client identifié par <vendor_id, client_type, client_version_major>. Dans le cas d'une **Image** de type image_target_id (identificateur d'image cible), le champ ECI_Image_Target_Id est fourni comme paramètre de requête. Au minimum, le serveur prendra en charge les **Fournisseurs** des **Clients ECI** servant à fournir leurs propres services d'**Opération de plate-forme**. Les **Hôtes ECI** utiliseront cette API pour s'efforcer d'acquérir les données de révocation les plus récentes relatives à l'ensemble des **Clients ECI** stockés.

```
client_revocation_data ::= 'client-revoc' '/' vendor_id .
```

- Cette requête renverra le fichier de données de révocation de **Client ECI** le plus récent d'un client identifié par **vendor_id**. Au minimum, le serveur prendra en charge les clients servant à fournir ses propres services d'**Opération de plate-forme**.

Tableau 7.7.3.5-1 – Syntaxe du fichier de version de client

Syntaxe	Nbre de bits	Mnémonique
ECI_Client_Version_File {		
magic = 'RCVE'	32	uimsbf
client_version	16	uimsbf
emergency_download_descriptor		
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RCVE'.
client_version: nombre entier	Version de client la plus récente du type de client identifié dans la requête.
emergency_download_descriptor	Descripteur **ECI_client_emergency_download_descriptor** dans lequel l'**Hôte ECI** supposera que l'étiquette d'opération de plate-forme correspondra à l'Opération de plate-forme du fournisseur de l'API web du client et le champ client_tag à l'image de client requise dans les paramètres de l'API web.

Tableau 7.7.3.5-2 – Syntaxe du fichier de version de justificatif d'identité de client

Syntaxe	Nbre de bits	Mnémonique
ECI_Client_Credential_Version_File {		
magic = 'RCCV'	32	uimsbf
root_version	8	uimsbf
vendor_rl_version	24	uimsbf
eci_vendor_id	32	uimsbf
padding(4)		
client_rl_version	24	uimsbf
eci_client_id	32	uimsbf
}		

Sémantique:

magic: octet [4]	Représentation ASCII 8 bits de la chaîne 'RCCV'.
root_version: nombre entier	Version racine (telle que définie dans le Tableau 5.3-1) du justificatif d'identité de **Client ECI** le plus récent.
vendor_rl_version: integer	Numéro de version de la liste de révocation du fournisseur de systèmes de sécurité du justificatif d'identité de **Client ECI** le plus récent.
eci_vendor_id: ECI_Vendor_Id	Identificateur de Fournisseur d'interface ECI (défini dans le Tableau 7.6.1-2) du justificatif d'identité de **Client ECI** le plus récent.
client_rl_version: integer	Numéro de version de la liste de révocation de client du justificatif d'identité de **Client ECI** le plus récent.
eci_client_id: ECI_Client_Series-Id	Identificateur de série de clients ECI (défini dans le Tableau 7.6.1-2) du justificatif d'identité de **Client ECI** le plus récent.

7.7.3.6 Requêtes de configuration du système de sécurité évoluée (AS_setup) de l'API web

Si l'**Opérateur** prend en charge l'enregistrement en ligne des **Clients ECI** en mode chiffrement, la requête suivante pourra être envoyée:

```
as_request ::= 'as_request' '/' vendor_id '/' eci_client_id
    '?&image-target-id=' target_id '&nonce=' nonce].
```

Cette requête renvoie le fichier as_setup pour le client spécifié (<vendor_id,eci_client_id>) et l'**Équipement CPE** spécifié par le champ ECI_Image_Target_Id target_id. Le type d'eci_client_id peut être ECI_Client_Id ou ECI_Client_Series_Id. 'Nonce' est la valeur du nonce spécifiée par le protocole de déchiffrement de l'**Image de client ECI**. Voir le § 7.8.4.2 pour obtenir des informations supplémentaires.

7.8 Installation d'un Client ECI d'opération de plate-forme

7.8.1 Domaine d'application et profil

L'**Opération de plate-forme** peut sélectionner les options de sécurité des installations de **Client ECI** et l'indiquer à l'aide du champ image_encrypted_flag et du drapeau en ligne dans le fichier d'**Image de client ECI** (voir le Tableau 7.6.1-2):

- Le "mode d'installation des **Clients ECI** avec un fichier d'**Image de client ECI** non chiffré" – dont le **Client ECI** (version la plus récente), tel que proposé par la signalisation définie dans le § 7.2, est téléchargé et le lancement du **Client ECI** a lieu.
- Le "mode d'installation des **Clients ECI** avec fichier d'**Image de client ECI** chiffré", qui, en sus du premier mode, permet à l'**Opération de plate-forme** de chiffrer et d'authentifier l'**Image de client ECI** comme défini dans la Recommandation [UIT-T J.1014] sur le déchiffrement des **Clients ECI**, est propre à l'**Hôte ECI** et comprend la vérification de la version de l'**Hôte ECI**, ce qui garantit la confidentialité des **Clients ECI** après déchiffrement puisque le déchiffrement n'est pas autorisé sur les **Hôtes ECI** inconnus ou compromis. Il faut une valeur ECI_Image_Target_Id si l'**Équipement CPE** n'est pas connecté à un réseau en ligne. Dans ce cas, ECI_Image_Target_Id doit être envoyé manuellement au serveur de tête de réseau chargé de la sécurité.

Le protocole des deux versions du lancement des **Clients ECI** est défini dans la suite de ce paragraphe.

Les **opérations de plate-forme** faisant fonctionner des **Équipements CPE** en ligne en mode installation avec chiffrement peuvent contraindre l'utilisation du **Client ECI** le plus récent en utilisant un nonce généré par le système de sécurité évoluée dans le protocole de déchiffrement avec le serveur de l'**Opération de plate-forme** du **Client ECI** (voir le § 7.7.3.6).

Règles de profilage:

- Si l'**Opération de plate-forme** propose l'enregistrement en ligne (la signalisation est définie dans le § 7.2) et que l'**Équipement CPE** peut accéder aux services en ligne, celui-ci utilisera le protocole d'enregistrement en ligne.
- Les **Équipements CPE** aptes à recevoir des programmes radiodiffusés pourront exécuter le protocole d'enregistrement en mode radiodiffusion. Ce mode requiert d'enregistrer l'**Équipement CPE** lors de l'enregistrement initial de l'**Opération de plate-forme**.
- Les opérations de plate-forme prenant en charge les réseaux de radiodiffusion acceptant les **Équipements CPE** sans connectivité en ligne simultanée pourront procéder à l'enregistrement en mode radiodiffusion. Les instructions relatives à l'entrée par l'**Utilisateur** des informations d'enregistrement d'un **Équipement CPE** respecteront les règles de formatage applicables.

7.8.2 Mode d'installation d'un Client ECI avec un fichier d'Image de client ECI non chiffré

Au début de l'initialisation du **Client ECI**, l'**Hôte ECI** réserve un **Créneau de sécurité évoluée** pour l'**Opération de plate-forme**, réinitialise ce créneau et charge la clé publique d'**Opération de plate-forme** dans le **Créneau de sécurité évoluée** comme défini dans la Recommandation [UIT-T J.1014].

Si nécessaire, l'**Hôte ECI** télécharge le **Client ECI**, le stocke dans la mémoire non volatile aux fins d'extraction future et le démarre. Le **Client ECI** guidera l'**Utilisateur** pendant l'installation. L'installation pourra demander à l'**Utilisateur** d'envoyer manuellement la valeur des champs ECI_image_Target_Id et target_id de l'**Équipement CPE** au serveur de tête de réseau si l'**Équipement CPE** est dépourvu d'une connexion en ligne permettant l'enregistrement à des fins de sécurité du système de radiodiffusion.

À chaque redémarrage ultérieur, l'**Hôte ECI** réinitialisera le **Client ECI**.

7.8.3 Mode d'installation d'un Client ECI avec un fichier d'Image de client ECI chiffré

Ce mode de fonctionnement utilise le téléchargement chiffré de l'**Image de client ECI** à l'aide d'une clé sélectionnée par l'**Opérateur**. Cette clé est chiffrée et se trouve dans une structure **as_setup**.

Au début du lancement du **Client ECI**, l'**Hôte ECI** réserve un **Créneau de sécurité évoluée** pour l'**Opération de plate-forme**, réinitialise ce créneau et charge la clé publique d'**Opération de plate-forme** dans le **Créneau de sécurité évoluée**:

- L'**Hôte ECI** distinguera deux modes d'extraction d'**as_setup**: **Mode enregistrement:** ce mode est adopté si le **Client ECI** est lancé pour la première fois, si la clé publique d'opération de plate-forme ou la version du **Client ECI** a changé ou si le client fonctionne en mode réenregistrement en ligne à l'aide d'un seul nonce par réenregistrement. La structure **as_setup** de l'**Équipement CPE** sera extraite du réseau de l'**Opération de plate-forme**.
- **Mode enregistré:** la structure **as_setup** précédente est extraite de la mémoire non volatile. En cas de changement imminent de version du **Client ECI** ou de l'**Hôte ECI**, le **Client ECI** doit demander à l'**Utilisateur** de lancer ou de débloquer le téléchargement (avec les paramètres de téléchargement par défaut, ces actions ont normalement lieu automatiquement dans un délai raisonnable). Le téléchargement d'un nouveau **Client ECI** nécessitera également une nouvelle structure **as_setup**.

En mode enregistrement, l'**Hôte ECI** effectuera les actions suivantes pour extraire une nouvelle structure **as_setup**:

1) L'**Hôte ECI** initialise le Créneau de sécurité évoluée et extrait:
   - la valeur ECI_Image_Target_Id et l'identificateur de cible (target_id) de l'**Équipement CPE**;
   - un nonce (128 bits) extrait du **Créneau de sécurité évoluée** par application de la fonction **getAsSlotRk** (voir la Recommandation [UIT-T J.1014]) en cas d'enregistrement en ligne.
2) L'**Hôte ECI** enverra les informations ci-dessus pour extraire un message **as_setup** de l'**Opération de plate-forme**:
   - En cas d'**enregistrement en mode radiodiffusion**, l'**Hôte ECI** présentera l'identificateur de cible à l'écran avec la boîte de dialogue d'enregistrement de l'**Opération de plate-forme**. L'**Hôte ECI** extraira la structure **as_setup** du carrousel de configuration du système de sécurité évoluée (voir le § 7.7.2).

   NOTE 1 – Si la plate-forme fournit plusieurs types de **Clients ECI**, l'**Opération de plate-forme** pourra demander à l'**Utilisateur** de fournir des informations supplémentaires afin de renvoyer la structure **as_setup** adaptée au type de **Client ECI** concerné.

   NOTE 2 – L'**Opération de plate-forme** pourra supposer que l'**Équipement CPE** a téléchargé la version la plus récente de l'**Image de client ECI** et ne fournir que la structure **as_setup** correspondant à cette image.

   - En cas d'**enregistrement en ligne**, l'**Équipement CPE** enregistrera l'identification du client, la valeur target_id de l'**Équipement CPE** et le nonce à l'aide de l'API web mentionnée au § 7.3.3.

   NOTE 3 – L'**Opération de plate-forme** peut décider d'appliquer le nonce pour renouveler l'enregistrement lors de chaque événement de réinitialisation de l'**Hôte ECI**.

Après la séquence d'acquisition de la structure **as_setup** en mode enregistrement ou la récupération de cette structure dans la mémoire non volatile en mode enregistré, l'**Hôte ECI** initialisera le système de sécurité évoluée et s'efforcera de charger le **Client ECI** chiffré;

1) de charger de la structure **as_setup** dans le système de sécurité évoluée à l'aide du message **reqAsClientImageDecrKey**; de charger la chaîne de certificats de **Clients ECI** dans le système de sécurité évoluée; de charger la liste de révocation des clients de l'**Opération de plate-forme** et la cosignature des clients de l'**Opération de plate-forme**. Les cas d'échec suivants doivent être au moins signalés à l'**Utilisateur** de façon intelligible, ou gérés automatiquement:

a) Ancienne version d'**Hôte ECI** – Il faut mettre à jour l'**Hôte ECI** ou son justificatif d'identité.

b) Ancienne version de **Client ECI** – Il faut mettre à jour le **Client ECI** ou son justificatif d'identité.

2) Déchiffrer l'image à l'aide de la clé d'**Image** de client calculée par le système de sécurité évoluée si nécessaire et authentifier l'**Image de client ECI** à l'aide de la signature de **Client ECI** et des cosignatures de l'**Opération de plate-forme**.

3) Échec en cas d'erreur de validation.

La structure as_setup et le format de fichier as_setup_file seront conformes à la définition donnée dans le Tableau 7.8.3-1.

Tableau 7.8.3-1 – Structure de configuration du système de sécurité évoluée, fichier et fichier de seau

Syntaxe	Nbre de bits	Mnémonique
ECI_As_Setup {		
as_version	8	uimsbf
if (as_setup_version == 0x01) {		
vendor_id	20	uimsbf
if (/* client image regular */){		
ECI_Client_id **client_id**		
}		
if (/* client image series */){		
ECI_Client_Series_Id **series_id**		
}		
ECI_Image_Target_Id **target_id**		
as_tag	16	uimsbf
online	1	uimsbf
padding(4)		
EciRootState **min_root_state**	32	
InputV inputV		
symKey eKey		
Extension **extension**		
}		
}		
ECI_As_Setup_File {		
magic file = 'AES'	24	uimsbf
as_setup_file_version	8	uimsbf
if (as_setup_version == 0x01){		
ECI_As_Setup **as_setup**		
}		
}		
ECI_As_Setup_Bucket_File {		
magic_bucket_file = 'AEB'	24	uimsbf
as_setup_bucket_version	8	uimsbf
if (as_setup_version == 0x01){ for (i=0; i<n; i++) {		
ECI_As_Setup **as_setup_item**		
}		
}		
}		

Sémantique:

vendor_id: nombre entier	Fournisseur de systèmes de sécurité du **Client ECI** auquel cette configuration du système de sécurité évoluée est destinée.
client_id: ECI_Client_Id	Identificateur du **Client ECI** auquel cette configuration du système de sécurité évoluée est destinée. La déclaration "if" précédente utilise le champ de type client_id: elle doit correspondre à "image normale du client".
series_id: ECI_Client_Series_Id	Identificateur de la **Série de Clients ECI** auquel cette configuration du système de sécurité évoluée est destinée. La déclaration "if" précédente utilise le champ de type client_id: elle doit correspondre à "série d'images de client".
target_id: ECI_Image_Target_id	Champ identifiant l'**Équipement CPE** auquel ce message est destiné.
client_tag: nombre entier	Étiquette indiquant la version de la structure as_setup pour la cible ci-dessus. La valeur doit changer en cas de modification de la structure **as_setup** de cette cible, par exemple, augmentation.
online: bool	Si la valeur de ce message est "true", il faut utiliser le nonce du créneau dans le mécanisme de clé d'authentification. Si elle est "false", aucun nonce n'est nécessaire. NOTE – Ce bit ne sera défini qu'en cas de connexion en ligne fonctionnelle.
min_root_state: minEciRootState	Etat Minimum Root (numéro minimum de version racine, numéro minimum de liste de révocation racine) à appliquer pour valider l'**Hôte ECI** et les **Clients ECI** chargés. Ce champ est codé sous forme de séquence d'octets comme défini dans la Recommandation [UIT-T J.1014].
inputV: InputV	Message InputV destiné au système de sécurité évoluée. Ce champ est codé sous forme de séquence d'octets comme défini dans la Recommandation [UIT-T J.1014].
eKey: SymKey	Clé symétrique chiffrée permettant de déchiffrer l'image. Ce champ est codé sous forme de séquence d'octets comme défini dans la Recommandation [UIT-T J.1014].
extension: Extension	Données d'extension, compatibles avec les versions antérieures. Ne doit pas dépasser 256 octets pour les applications de radiodiffusion afin de maintenir la compacité des carrousels de radiodiffusion. Aucune application n'est définie pour ces données.
magic_file: octet[3]	Représentation ASCII 8 bits de la chaîne 'AES'.
as_setup_file_version: integer	Version du format de ECI_AS_Setup_File. Les valeurs 0 et 0x2..0xff sont réservées. Le format défini ici utilise la valeur 0x01.
as_setup: ECI_As_Setup	Structure **as_setup** de l'**Opération de plate-forme** permettant de charger un **Client ECI** chiffré particulier sur un **Hôte ECI** donné.
magic_bucket_file: octet[3]	Représentation ASCII 8 bits de la chaîne 'AEB'.
as_setup_item: ECI_As_Setup	Structures as_setup de ce seau. Toute nouvelle structure as_setup sera ajoutée en haut du seau; la plus ancienne structure as_setup se trouvera donc au fond du seau. Des structures as_setup ne seront supprimées, si nécessaire, qu'à partir du fond du seau. Cette approche accélère l'inspection des mises à jour par les **Équipements CPE**. En effet, après la première vérification, on a besoin uniquement de vérifier les structures as_setup de haut en bas jusqu'à ce qu'on rencontre la première des séries précédemment vérifiées.

La fréquence minimale de vérification des mises à jour de la structure **as_setup** sera identique à celle utilisée pour les autres données des **Clients ECI** définie dans le § 7.3.1. À noter qu'en général, la mise à jour désigne la mise à jour du logiciel du **Client ECI** et/ou de l'**Hôte ECI** de l'**Équipement CPE.** Par conséquent, leur mise à jour sera également téléchargée afin d'assurer la cohérence de la séquence d'initialisation des **Clients ECI**. Si un nouvel ensemble cohérent de ce type n'est pas disponible, on peut utiliser l'ensemble cohérent précédent.

L'**Hôte ECI** qui procède à l'enregistrement en mode radiodiffusion (manuel) d'un **Client ECI** nouveau ou mis à jour vérifiera le plus fréquemment possible s'il existe une mise à jour du carrousel de fichiers as_setup.

7.8.4 Protocole de transport

7.8.4.1 Protocole de radiodiffusion

Le protocole de radiodiffusion des structures **as_setup** sera conforme au § 7.7.2.

Le nombre de structures as_setup à mettre à jour lors du changement de version d'un **Client ECI** peut être très important. Afin de limiter le nombre de nouveaux messages as_setup en ligne lors du changement de version d'un **Client ECI** dans le cadre d'une opération de grande ampleur en mode radiodiffusion seulement, l'**Opération de plate-forme** pourra mettre à disposition un nouveau **Client ECI** et *organiser* la restitution de nouveaux justificatifs d'identité, ce qui remplacera des groupes de **Clients ECI** sur les **Équipements CPE**. Elle pourra réitérer cette procédure plusieurs fois afin de toucher un maximum d'**Équipements CPE** avant de recourir au système de sécurité pour lancer l'utilisation du nouveau **Client ECI**.

7.4.8.2 Protocole en ligne

Le protocole en ligne s'appuie sur un protocole simple de requête-réponse entre l'**Équipement CPE** et le **Client ECI**, défini au § 7.7.3, qui consiste à transmettre la valeur **target_id** de l'**Équipement CPE** et le **nonce** dans la requête et à renvoyer le fichier **ECI_AS_Setup_File**.

7.8.5 Présentation de l'identificateur de cible à l'Utilisateur

En l'absence de connexion en ligne, **l'Hôte ECI** et le **Client ECI** doivent pouvoir présenter l'identificateur de cible (target_id) de l'**Équipement CPE** à l'**Utilisateur** sur les réseaux de radiodiffusion, afin de générer les informations propres à l'**Équipement CPE** requises pour déchiffrer l'**Image de client ECI**, si nécessaire, ainsi que des messages InitV du système de sécurité évoluée du **Client ECI** (le protocole de transport de ces messages est défini par le **Client ECI**). L'identificateur de cible doit également pouvoir être lu sur un élément imprimé apposé à l'extérieur de l'**Équipement CPE** ou figurant dans sa documentation. Le présent paragraphe définit la présentation de l'identificateur de cible à l'**Utilisateur**.

La valeur de target_id est un nombre entier de 64 bits. Sa présentation à l'**Utilisateur** sera conforme aux règles énoncées dans le § 6.2.2. Elle fera appel à une somme de contrôle de 9 bits et ajoutera des chaînes secondaires de 9 bits au lieu de 5 bits. De ce fait, target_id est représenté par une séquence de six nombres à 4 chiffres compris entre 0 et 7.

Les **Équipements CPE** et les **Clients ECI** sont autorisés à utiliser des représentations personnalisées dans leur interface d'**Utilisateur** (par exemple, basées sur un système privé de numérotation des **Équipements CPE**), mais disposeront toujours de fonctions d'enregistrement des **Clients ECI** fondées sur le format de présentation ci-dessus.

8 Révocation

8.1 Introduction

Toutes les parties et tous les éléments contribuant à l'**Écosystème ECI** seront certifiés par l'**Autorité de confiance ECI**. Cette certification garantira la qualité de base des fonctionnalités ainsi que la fiabilité des mises en œuvre et permettra aux participants de prendre les mesures de renouvellement appropriées. Ce processus de certification empêche également tout piratage à l'aide de l'écosystème de l'interface **ECI**.

L'interface **ECI** possède des fonctions de refus sélectif de fourniture de services aux **Équipements CPE** en fonction du statut attribué par l'**Autorité de confiance ECI** au matériel **CPE**, à l'**Hôte ECI**, aux autres **opérations de plate-forme** et aux **Clients ECI** chargés.

L'**Autorité de confiance ECI** peut révoquer une **Opération de plate-forme** si celle-ci ne respecte pas les règles généralement admises, notamment celles relatives à la non-interférence avec d'autres **opérations de plate-forme** sur des **Équipements CPE** partagés ou à la fourniture de services pirates par le biais de l'interface **ECI**. De la même manière, elle peut révoquer les **Clients ECI** qui ne respectent pas les règles communément admises, notamment la non-interférence avec d'autres **Clients ECI** sur des **Équipements CPE** partagés ou les pratiques de piratage. Elle peut aussi révoquer les

versions du logiciel de l'**Hôte ECI** présentant des failles importantes qui révèlent les secrets des **Clients ECI** ou rendent possibles les manipulations.

Dans tous les cas ci-dessus, les organisations responsables de l'élément révoqué peuvent remédier au problème, en général en le remplaçant par un nouvel élément. Un **Fournisseur de systèmes de sécurité** peut remplacer un **Client ECI** par une nouvelle version, un **Fabricant d'équipement CPE** peut fournir des correctifs de sécurité pour un **Hôte ECI** et un **Opérateur** peut améliorer ses opérations grâce à une nouvelle version de son **Certificat d'opération de plate-forme**. Toutes ces opérations étant collaboratives par nature, il est suggéré d'en faire l'objet d'accords contractuels entre les parties concernées et l'**Autorité de confiance ECI**.

Si les parties impliquées dans l'interface **ECI** contreviennent systématiquement aux accords passés avec l'**Autorité de confiance ECI** et, ce faisant, portent préjudice à d'autres parties ou aux **Utilisateurs**, l'**Autorité de confiance ECI** est en droit de révoquer tous les éléments qui leur sont imputables.

Les **Équipements CPE** dont l'**Hôte ECI** n'est plus valable, et pour lesquels leur **Fabricant** ne prévoit pas de mises à jour, peuvent être révoqués. Il en va de même si l'amorceur de l'**Équipement CPE** est compromis et autorise le chargement de logiciels d'**Hôte ECI** non conformes.

Les **Équipements CPE** s'efforceront de remplacer automatiquement une version révoquée par une version mise à jour, s'il en existe une. Cependant, les nouveaux téléchargements et les **Listes de révocation** pourront être bloqués. Dans ce cas, l'**Opération de plate-forme** pourra refuser la prestation des services ou refuser la restitution des contenus stockés localement sur l'**Équipement CPE** concerné.

8.2 Révocation des Équipements CPE

L'interface **ECI** autorise les **opérations de plate-forme** à utiliser la fonction de fourniture sélective des droits du système CA ou DRM pour refuser la fourniture de services à certains **Équipements CPE**. L'**Opération de plate-forme** peut examiner l'état le plus récent attribué par l'**Autorité de confiance ECI** à un **Équipement CPE**. Si l'**Autorité de confiance ECI** juge nécessaire de révoquer un **Équipement CPE**, l'**Opération de plate-forme** peut désactiver la fourniture des services à l'équipement concerné sur la base de son identificateur de puce électronique enregistré auprès du système CA ou DRM fournissant les services.

La présente Recommandation permet également aux **opérations de plate-forme** d'exclure la fourniture de services aux **Équipements CPE** exécutant des **Hôtes ECI** révoqués. Elles peuvent utiliser le système de sécurité évoluée pour exiger un numéro de version minimum de l'**Hôte ECI** conforme à une liste de révocation d'**Hôte ECI** récente, comme défini au § 8.3.

Le mécanisme de révocation des **Hôtes ECI** peut également révoquer des **Équipements CPE**, le cas échéant, en spécifiant une version minimum d'**Hôte ECI** supérieure à celle utilisée jusqu'alors.

8.3 Processus de révocation générique

Ce paragraphe porte sur la combinaison d'une version minimum de **Racine** et d'une version minimum de liste de révocation **Racine** afin de constituer une "version minimum de **Liste de révocation**".

L'insuffisance des services constitue le mécanisme ultime de révocation d'un **Hôte ECI**: si un élément révoqué est présent sur l'**Hôte ECI** en dépit de l'application de **Listes de révocation** (supposément anciennes), l'**Opération de plate-forme** peut décider de cesser la fourniture des services à l'**Hôte ECI** en question. La fourniture d'une liste de révocation minimum acceptable requise par une **Opération de plate-forme** est protégée par le **Système de sécurité évoluée**: sa manipulation suffira à déclencher l'insuffisance des services. Une **Opération de plate-forme** peut ainsi contraindre la vérification de la version du justificatif d'identité utilisé pour installer l'**Hôte ECI**, ainsi que toutes les autres **opérations de plate-forme** et tous les **Clients ECI**.

L'**Opération de plate-forme** fournira un service de téléchargement de la **Liste de révocation** de l'un quelconque des éléments ci-dessus (**Hôtes ECI**, **Clients ECI** et **opérations de plate-forme**), ce qui garantit la disponibilité des listes de révocation les plus récentes de tous les **Clients ECI** et de toutes les opérations de plate-forme chargés sur l'**Hôte ECI**.

L'initialisation du **Système de sécurité évoluée** (Recommandation [UIT-T J.1014]) permet à l'**Hôte ECI** de spécifier cette version minimum attendue des **Listes de révocation** pour tous les éléments. Elle sert à valider rétrospectivement la version de la liste de révocation utilisée par l'**Hôte ECI**. Celui-ci utilisera la valeur minimum de la **Liste de révocation racine** des **éléments** du **Client ECI** qu'il souhaite charger et l'**Image d'hôte ECI** qu'il a chargée.

NOTE – Il est suggéré à l'**Hôte ECI** de ne pas charger les éléments susceptibles d'entraîner une révocation, mais plutôt de prévenir l'**Utilisateur**.

Afin d'éviter une insuffisance des services infondée, il faut que les justificatifs d'identité les plus récents (et, si nécessaire, les versions les plus récentes) de tous les éléments à charger soient disponibles sur l'**Hôte ECI**. Pour empêcher un dysfonctionnement des **Clients ECI** du fait des risques pour la sécurité liés à la présence d'**Hôtes ECI**, de **Certificats d'opération de plate-forme** ou de **Clients ECI** révoqués, l'**Hôte ECI** fournira la fonctionnalité suivante afin d'assurer la disponibilité des justificatifs d'identité et (si nécessaire) des éléments les plus récents et d'éviter les conséquences d'une insuffisance des services injustifiée:

- Elle conservera la chaîne de **Listes de révocation de** l'**Autorité de confiance ECI** la plus récente vérifiée dans sa configuration actuelle d'**Hôte ECI**, d'**Opération de plate-forme** et de **Clients ECI** à l'aide des services de téléchargement des justificatifs d'identité et des **Listes de révocation** du **Fabricant de l'Équipement CPE** et de l'**Opération de plate-forme** de ses **Clients ECI**.
- Les paramètres par défaut de tous les modes des **Équipements CPE** pertinents permettront ce téléchargement.

  L'**Équipement CPE** ne sera pas doté d'un mode de fonctionnement empêchant en permanence le téléchargement, hormis la mise hors tension du matériel ou le blocage de l'accès au réseau de téléchargement (non imputable à un état ou à un mode de fonctionnement de l'**Équipement CPE**).
- Une action simple de l'**Utilisateur** permettra de restaurer les paramètres par défaut du téléchargement et de la révocation des **Clients ECI** et des **opérations de plate-forme**.

La présente Recommandation permet aux **Utilisateurs** d'ignorer le comportement par défaut de l'Hôte pour révoquer des éléments entraînant l'insuffisance des services pour d'autres hôtes. S'ils le font (par exemple, pour qu'un Client ancien continue à s'exécuter), ils risquent d'avoir de plus en plus de mal à rendre des services d'actualité.

8.4 Révocation d'Hôte ECI fondée sur des Listes de révocation

L'**Hôte ECI** d'un **Équipement CPE** mal géré peut s'avérer révoqué. Les **Fabricants d'Équipements CPE** doivent fournir des justificatifs d'identité mis à jour, notamment la **Liste de révocation ECI** applicable la plus récente. De plus, une **Opération de plate-forme** envisageant d'exécuter un **Client ECI** sur un **Hôte ECI** peut fournir un service de téléchargement pour une **Liste de révocation** relative aux justificatifs d'identité de l'**Hôte ECI** ainsi que pour certains **Hôtes ECI**. L'**Hôte ECI** appliquera les **Listes de révocation** des justificatifs d'identité des **Hôtes ECI** (**Certificat racine** et **Certificat de fabricant**) conformément aux règles de traitement des **Listes de révocation** génériques définies dans la Recommandation [UIT-T J.1014].

Le format du fichier de données de révocation de l'**Hôte ECI** est défini au § 5.3.

8.5 Révocation des opérations de plate-forme ECI

Une **Opération de plate-forme** envisagant d'exécuter un **Client ECI** sur un **Hôte ECI** peut fournir un service de téléchargement pour une **Liste de révocation** relative aux justificatifs d'identité d'autres **opérations de plate-forme**. L'**Hôte ECI** appliquera les **Listes de révocation** à tous les justificatifs d'identité des **opérations de plate-forme** installées, conformément aux règles de traitement des **Listes de révocation** génériques définies dans la Recommandation [UIT-T J.1014].

Le format du fichier de révocation des **opérations de plate-forme ECI** est défini au § 7.6.3.

8.6 Révocation des Clients ECI

Une **Opération de plate-forme** envisageant d'exécuter un **Client ECI** sur un **Hôte ECI** peut fournir un service de téléchargement pour une **Liste de révocation** relative à d'autres **Clients ECI**. L'**Hôte ECI** appliquera les **Listes de révocation** à tous les justificatifs d'identité des **Clients ECI** installés, conformément aux règles de traitement des **Listes de révocation** génériques définies dans la Recommandation [UIT-T J.1014].

Le format du fichier de révocation des **Clients ECI** est défini au § 7.6.3.

9 Interfaces des Clients ECI

9.1 Introduction

9.1.1 Architecture des interfaces des Clients ECI

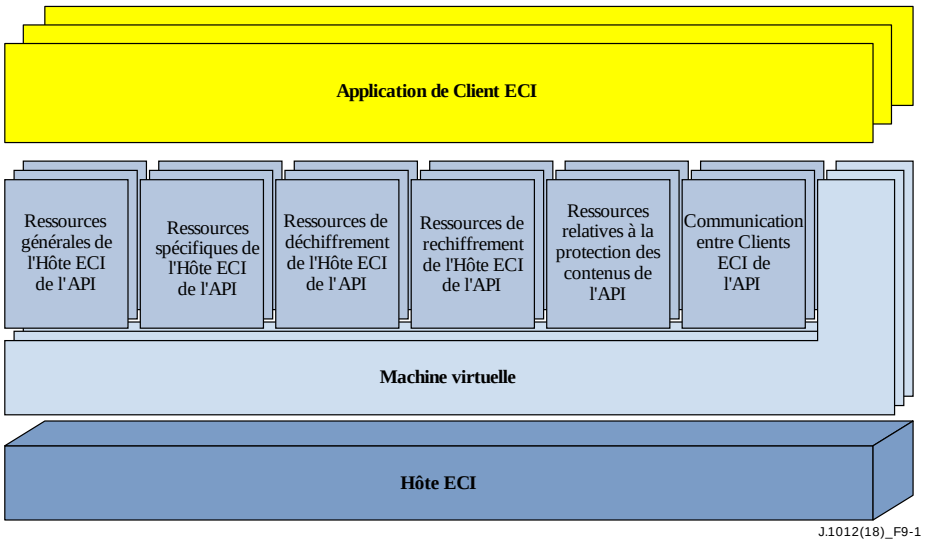


Figure 9.1.1-1 – Structure des API définies au § 9

La Figure 9.1.1-1 donne une vue d'ensemble de la structure des API du système ECI. Elle montre 6 blocs d'API que les **Clients ECI** peuvent utiliser. Ces blocs sont spécifiés aux § 9.4 à 9.9. Le Tableau 9.1.1-1 fournit la liste des API définies au § 9 de la présente Recommandation. Voir également [b-ETSI GS ECI 002].

Tableau 9.1.1-1 – liste des API définies dans la présente Recommandation

Paragraphe N°	Catégorie d'API	Description
9.4	API relatives aux ressources générales de l'**Hôte ECI**	API prenant en charge les fonctionnalités générales des **Clients ECI**
9.5	API relatives aux ressources propres à l'interface **ECI** de l'**Hôte ECI**	API prenant en charge les fonctionnalités propres aux **Clients ECI**
9.6	API relatives à l'accès aux ressources de déchiffrement de l'**Hôte ECI**	API autorisant les **Clients ECI** à utiliser les ressources de déchiffrement de l'**Hôte ECI**
9.7	API relatives à l'accès aux ressources de rechiffrement de l'**Hôte ECI**	API autorisant les **Clients ECI** à utiliser les ressources de rechiffrement de l'**Hôte ECI**
9.8	API relatives aux ressources en rapport avec les propriétés de contenu	API prenant en charge les fonctionnalités de protection des contenus des **Clients ECI**
9.9	API relatives à la communication entre **Clients ECI**	API prenant en charge la communication directe entre **Clients ECI**

9.1.2 Pointeur de média

Un **Pointeur de média** est un identificateur d'objet situé dans l'environnement de l'hôte fournissant le contexte pour toutes les interfaces d'**Hôte ECI** fournies aux **Clients ECI** afin de contrôler le processus de déchiffrement d'un élément de contenu. Il permet également aux **Clients ECI** de spécifier les données du conteneur de contenus dont ils ont besoin pour désembrouiller les contenus. En cas de fourniture par un réseau de radiodiffusion, il offre aussi une fonctionnalité de contrôle de la sélection du programme à décoder et du flux du réseau de livraison (fonction de syntonisation). Un **Client ECI** peut également demander un **Pointeur de média** doté d'un accès à un syntoniseur afin d'accéder aux données requises pour son exécution à partir des flux de réseau auxquels l'application/l'hôte n'accède pas à des fins d'acquisition de contenus. Si la fourniture est fondée sur des fichiers ou des flux OTT, le **Pointeur de média** permet au **Client ECI** d'accéder à des données de sécurité dans le fichier/flux non spécifiées dans un emplacement normalisé.

Le désembrouillage de la session de média s'opère sous le contrôle direct du **Client ECI**. La synchronisation de l'application d'un mot de contrôle et du flux de transport repose sur les informations de commande d'embrouillage du flux de transport. La synchronisation de mots de contrôle (couramment appelés clés dans ce contexte) et d'un fichier ISO BMFF CENC [ISO/CEI 23001-7] reposera sur des identificateurs KeyID CENC.

Les sessions ayant recours à un **Pointeur de média** sont présentées dans le Tableau 9.1.2-1.

Tableau 9.1.2-1 – Types de Pointeurs de média

Nom	Valeur	Description
MhDvbTs	0x01	Le flux de transport sera conforme à [ISO/CEI 13818-1-1].
MhIsobmffCenc	0x10	Le fichier ISO BMFF sera conforme à [ISO/CEI 23001-9] et [ISO/CEI 14496-12].
RFU	autre	Réservé à une utilisation future (Reserved for future use)

9.2 Interface de Machine virtuelle ECI

9.2.1 Principes

Une instance distincte de machine virtuelle sera créée pour chaque **Client ECI**. Le chargement des données et des instructions de transformation d'un **Client ECI** en machine virtuelle est défini au § 7.

Le fonctionnement de la machine virtuelle est défini dans la Recommandation [UIT-T J.1013]. Voir également [b-ETSI GS ECI 001-4].

Toute interaction du **Client ECI** avec le monde extérieur passera par l'interface de messages définie au § 9.2.3.

9.2.2 Instructions et données (ressources statiques)

La machine virtuelle exécutera les instructions du **chargeur de Client ECI** à son intention figurant dans le(s) segment(s) de code de l'**Image de client ECI**.

La machine virtuelle empêche l'auto-modification des instructions. Tout code qui générerait facilement un comportement indésirable et/ou aisément manipulable chez un **Client ECI** (par exemple, interpréteurs) est considéré inapproprié et doit être repéré dans le cadre du processus de certification des **Clients ECI**.

Un code et un espace de données statiques maximum requis par un **Client ECI** sont proposés dans le document [b-UIT-T J Suppl. 7].

9.2.3 Interaction avec l'Hôte ECI

La définition de toutes les interactions du **Client ECI** avec l'**Hôte ECI** repose sur le modèle de message présenté dans ce paragraphe. Les seules données partagées par le **Client ECI** et l'**Hôte ECI** sont les suivantes:

- données contenues dans les messages;
- données stockées dans la mémoire non volatile de l'**Hôte ECI** pour le compte du **Client ECI** ou
- données contenues dans les messages des canaux de communication en provenance ou à destination d'autres **Clients ECI**.
  À noter que ces données sont également échangées par le biais de messages.

Le modèle de message repose sur trois types d'échanges entre le **Client ECI** et l'**Hôte ECI**:

1) Échange initié par un **Client synchrone**: le **Client ECI** appelle une **Fonction de l'Hôte ECI** qui réagit très rapidement. Le fil du **Client ECI** (flux d'exécution) est bloqué pendant que l'**Hôte ECI** traite le message et y répond.
2) Échange initié par un **Client asynchrone**: le **Client ECI** envoie à l'**Hôte ECI** un message de **Requête de client** qui sera mis en file d'attente et traité le moment venu par l'**Hôte ECI**. L'appel asynchrone enverra un message de **Renvoi** immédiat ne contenant qu'un résultat basique (identificateur de message ou erreur). L'**Hôte ECI** produira ultérieurement un message de **Réponse de l'hôte** renvoyant le statut et les résultats de l'opération de l'**Hôte ECI** initiée par le **Client ECI**.
3) Échange initié par un **Hôte asynchrone**: l'**Hôte ECI** envoie au **Client ECI** un message qui sera mis en file d'attente et traité le moment venu par le **Client ECI**. L'appel asynchrone enverra un message de renvoi immédiat ne contenant qu'un résultat basique (standard). Le type et le format de ce message tel que représenté dans l'**Hôte ECI** ne relèvent pas de la présente Recommandation car il s'agit d'une question interne à l'**Hôte ECI**:

   Il est à noter que seule la représentation relative au client ECI est définie. Le client ECI enverra ultérieurement un message de Réponse contenant le statut et les résultats de l'opération du client ECI initiée par l'hôte ECI.

Les différents types d'échange de messages entre un hôte ECI et un client ECI sont représentés dans la Figure 9.2.3-1.

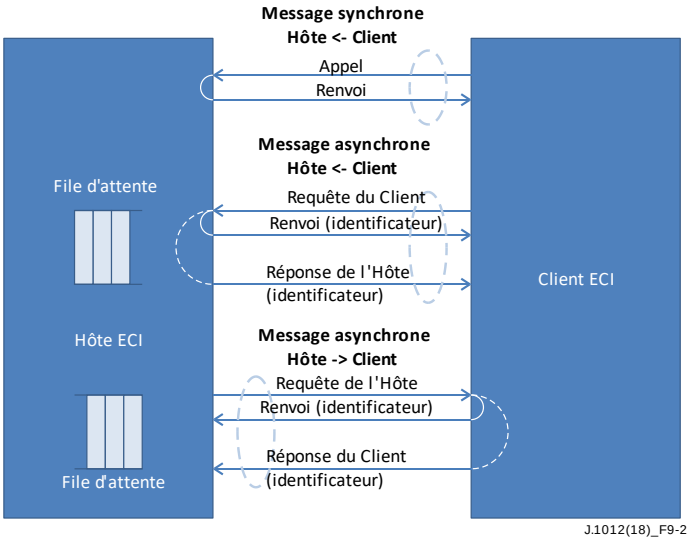


Figure 9.2.3-1 – Échanges de messages entre Client et Hôte

Le **Client ECI** doit s'assurer que les données utiles sont protégées comme il convient, par exemple à l'aide de mots de contrôle et de propriétés du contenu. De plus, l'interface n'est pas conçue pour les échanges de contenus.

Le **Client ECI** mettra en œuvre des Réponses aux **Requêtes de l'Hôte ECI** qu'il prend en charge, conformément aux définitions des API présentées au § 9 et utilisera pour ce faire l'identificateur des **Requêtes** dans la **Réponse**.

L'**Hôte ECI** mettra en œuvre des Réponses aux **Requêtes du Client ECI** qu'il prend en charge, conformément aux définitions des API présentées au § 9 et utilisera pour ce faire l'identificateur des **Requêtes** dans la **Réponse**.

Une **Requête** asynchrone pourra éventuellement indiquer qu'aucune **Réponse** n'est requise, par exemple, en cas de déplacement de nombreux éléments de données, si l'initiateur ne demande de **Réponse** qu'à la dernière **Requête** en partant du principe que tous les éléments de données intermédiaires sont traités correctement.

Toutes les **Requêtes de l'Hôte ECI** asynchrones et toutes les **Réponses de l'Hôte ECI** sont mises en file d'attente "dans leur ordre d'occurrence".

9.2.4 Ressources dynamiques fournies aux Clients ECI

Des paramètres techniques des ressources dynamiques minimum requises d'un **Client ECI** sont proposés dans le document [b-UIT-T J Suppl. 7]. Les éléments suivants sont abordés: fils, espace de pile, segments de mémoire, temps d'exécution, stockage dans la mémoire non volatile et communication entre les clients.

9.2.5 Gestion de la version des API

Les API définies dans la présente Recommandation peuvent avoir plusieurs versions, par exemple afin de proposer des fonctionnalités améliorées pour remplacer des fonctionnalités précédentes ou de combler des lacunes dans les spécifications. Lors de l'initialisation, les **Clients ECI** et leur **Hôte ECI** doivent déterminer les API à prendre en charge par leur homologue et sélectionner la version de chaque API disponible de l'homologue à utiliser jusqu'à la fin du cycle de vie des **Clients ECI**. Les **Clients ECI** ne peuvent utiliser que les API de découverte pendant la phase d'initialisation car les versions des messages (c.-à-d. leur disponibilité, leur longueur et leur syntaxe) ne sont définies qu'à la fin du processus de découverte.

Les versions des API figurent dans leur sémantique: l'interaction des messages entre un **Client ECI** et un **Hôte ECI** via une version d'API ne dépend ni de la prise en charge d'autres versions de cette API dans l'**Hôte ECI**, ni des interactions de l'**Hôte ECI** avec d'autres **Clients ECI** utilisant d'autres versions de cette même API.

NOTE 1 – Pour des raisons pratiques, le texte des paragraphes définissant les nouvelles versions des API peut faire référence aux définitions d'anciennes versions figurant dans la présente Recommandation.

Les API sont obligatoires, facultatives ou conditionnelles (c.-à-d. obligatoires sous certaines conditions). Le fait que les API en rapport avec les enregistreurs vidéo personnels (PVR) nécessitent un **Équipement CPE** prenant en charge cette fonctionnalité en constitue un exemple. Les futures versions de la présente Recommandation pourront définir le profil des API que les **Hôtes ECI** et les **Clients ECI** doivent prendre en charge en fonction du nom de profil et du numéro de version de la spécification.

Afin de respecter les dispositions de la présente Recommandation et d'assurer la compatibilité avec les versions antérieures, les **Hôtes ECI** et les **Clients ECI** prenant en charge une API prendront en charge toutes ses versions (y compris la plus récente) sauf si les anciennes versions sont explicitement déclarées obsolètes dans la présente Recommandation (et ses futures versions) ou d'une autre manière.

NOTE 2 – La création d'une version future de la présente Recommandation n'implique pas que les **Clients ECI** et les **Hôtes ECI** déployés ou nouveaux doivent y être conformes. Les politiques de mise à jour sur site des **Hôtes ECI** et des **Clients ECI** visant à respecter les nouvelles versions des spécifications ou des règles rendant obligatoire de nouvelles versions des spécifications, qui s'appliquent aux nouveaux **Hôtes ECI** et **Clients ECI**, ne relèvent pas du champ d'application de la présente Recommandation.

Les **Clients ECI** doivent sélectionner le numéro de version le plus élevé des API disponibles dans les **Hôtes ECI** qu'ils sont capables de gérer; et inversement, les **Hôtes ECI** doivent sélectionner le numéro de version le plus élevé disponible des API des **Clients ECI** qu'ils sont capables de gérer. Cette règle incite à migrer les versions vers des API plus mûres et évite les problèmes de compatibilité en cas d'abandon de versions (anciennes) des API.

Compte tenu du cycle de vie généralement plus long des **Hôtes ECI** et de la relative facilité de mise à jour des **Clients ECI**, ces derniers devraient être en mesure de prendre en charge des versions anciennes des API de l'**Hôte ECI** correspondant à la situation de la base installée (ce qui peut faire l'objet d'autres accords ne relevant pas de la présente Recommandation). A l'inverse, les nouveaux **Hôtes ECI** devraient prendre en charge les **Clients ECI** anciens correspondant au déploiement des **Clients ECI** (ce qui peut faire l'objet d'autres accords ne relevant pas de la présente Recommandation).

L'API de découverte entre **Clients ECI** et **Hôtes ECI** est définie au § 9.4.2.

9.2.6 Suivi de la capacité de réponse

L'**Hôte ECI** déploiera des fonctions de base de redémarrage automatique des **Clients ECI** afin de renforcer la fiabilité des fonctionnalités globales de l'**Équipement CPE**. Il détectera les conditions d'erreurs fatales dans les **Clients ECI** et réinitialisera automatiquement ces derniers lorsqu'elles se produiront. Toutes les ressources utilisées par les **Clients ECI** seront libérées avant la réinitialisation, y compris les **Pointeurs de média**, les sessions d'interface homme-machine (MMI), les fichiers, les connexions IP, etc.

Les conditions d'erreur suivantes sont définies:

- L'**Hôte ECI** surveillera l'exécution d'éventuelles instructions illicites par le code du **Client ECI**, telles que opcode d'instructions non défini, adressage de données illicites ou d'un code inexistant, dépassement ou insuffisance des piles de registre, etc.
- L'acceptation d'un nouveau message par le **Client ECI** sera soumise à un délai d'expiration par l'**Hôte ECI**. Un chiffre est proposé pour ce paramètre dans le document [b-UIT-T J Suppl. 7].

En cas de réinitialisations répétées, l'**Hôte ECI** pourra utiliser une politique, susceptible de nécessiter des paramètres **Utilisateur** ou l'entrée de données par l'**Utilisateur**, visant à décoder et à exclure de façon plus permanente le **Client ECI** fréquemment défaillant.

NOTE – L'exécution d'une commande syscall sys_exit (voir la Recommandation UIT-T J.1013]) par un **Client ECI** sera comprise comme la fin normale d'exécution de ce client. En général, elle signifie que le **Client ECI** peut être supprimé ou qu'il est remplacé par une version plus récente. Ce type d'événement n'entraîne pas la suppression automatique du **Client ECI** par l'**Hôte ECI**. En revanche, celui-ci attend qu'une procédure appropriée de remplacement ou de suppression soit invoquée par le biais d'autres politiques de gestion des **Clients ECI**.

9.3 Mécanisme des API des Clients ECI

9.3.1 Syntaxe des messages asynchrones

Toutes les structures des messages sont définies en fonction de leur apparition dans la machine virtuelle **ECI**. Le Tableau 9.3-1 présente la structure des tampons de tous les messages asynchrones en fonction de leur apparition dans la carte de la mémoire de la machine virtuelle. À noter que tous les tampons des messages sont alignés sur une frontière de 32 bits.

Tableau 9.3-1 – Syntaxe des messages asynchrones

Syntaxe dans le style du langage C	Nbre de bits
struct messageBuffer {	
uint32 **msgTag**;	32
uint16 **msgId**	16
uint16 **payloadLen;**	16
uint32 **payload[];**	n*32
} MessageBuffer;	

msgTag:

Ce champ représente les valeurs suivantes:

- Bits 0-15: **msgApiTag**. Identification de l'API du message (pour la définition, voir l'Annexe C).
- Bits 16-23: **msgCallTag**. Identification de l'appel API que le récepteur doit interpréter dans le contexte de la valeur de **msgTag** et de la version convenue de l'API.
- Bits 24-31: **msgFlags**: Drapeaux supplémentaires qualifiant un message. Les définitions suivantes s'appliquent:
  - Bit 24: **msgNoResFlag**: pour les messages de type **Requête** et demande: si 0b1, aucune **Réponse** n'est requise; si 0b0, une **Réponse** est requise. Ce bit n'a pas de signification dans les messages de réponse.
  - Les bits 25-31 sont réservés à une utilisation future. Ils seront mis à 0b0 par l'auteur du message.

L'étiquette des messages sera identique pour les **Réponses** aux messages de **Requête** et de demande.

msgId:

- Valeur de l'identificateur du message attribuée par l'**Hôte ECI**. Pour un message de réponse, ceci correspond à la valeur du message de requête d'origine. Le **Client ECI** envoyant la requête peut ne pas initialiser ce champ (la valeur sera attribuée par l'**Hôte ECI** et renvoyée sous forme de valeur de résultat de la commande syscall SYS_PUTMSG).

payloadLen:

- Champ de longueur des données utiles représentant la taille du tampon des données utiles en octets. La taille effectivement allouée au champ de données utiles sera cette valeur arrondie

au multiple de 4 (ou plus) suivant. Lors de l'interprétation du champ **payload** d'un message reçu, les **Hôtes ECI** vérifieront que les données n'excèdent pas la valeur de **payloadLen**, sinon ils renverront une erreur. Les **Clients ECI** peuvent partir du principe que les **Hôtes ECI** fournissent des tampons de messages de dimension correcte.

champ payload:

- Le champ payload contient les paramètres des messages. Sa structure est définie à l'aide de la syntaxe en langage C de la signature de l'appel de fonction utilisée avec les règles de mappage particulières définies au § 9.3.2.3.

9.3.2 Convention de définition du format des messages asynchrones

9.3.2.1 Syntaxe de la définition des messages

Les messages asynchrones sont définis à l'aide d'une déclaration de signature de fonction, sur le modèle du langage C. Cette notation correspond au format des messages prescrit par les règles définies dans le présent paragraphe. Un exemple de déclaration de signature de fonction figure ci-dessous:

```
reqSetTimer(uint32 time, uchar priority)
```

9.3.2.2 Types de paramètres de base des messages

La syntaxe utilisera les types de base de définition des paramètres spécifiés dans le Tableau 9.3.2.2-1.

Tableau 9.3.2.2-1 – Types de base utilisés pour définir les paramètres des messages

Types de base	Représentation
uint8, uchar, byte:	Nombre entier de 8 bits non signé
int8, char, bool:	Nombre entier de 8 bits signé
uint16, ushort:	Nombre entier de 16 bits non signé
int16, short:	Nombre entier de 16 bits signé
uint32, uint:	Nombre entier de 32 bits non signé
int32, int:	Nombre entier de 32 bits signé
uint64, ulong:	Nombre entier de 64 bits non signé
int64, long:	Nombre entier de 64 bits signé
char *, … ,long * (mémoire du client)	32 bits; autorisé uniquement pour les messages synchrones

Pour les paramètres de type booléen, les valeurs symboliques **True** et **False** sont utilisées. Conformément à la définition du langage C, **False** est représenté par 0x00, **True** par toute autre valeur.

9.3.2.3 Mappage des données utiles des messages aux paramètres des messages

Le champ **payload** contient tous les paramètres du message. Le paramètre identificateur des messages **msgId** et les paramètres de résultat **msgResult** sont implicites au sens où ils ne figurent pas dans la description de la syntaxe déclarative de la signature de fonction. Leur présence est implicitement définie par le type de message.

L'**Hôte ECI** associera un champ **msgId** aux messages de **Requête** de l'**Hôte ECI** et du **Client ECI** afin d'associer la **Requête** à la réponse correspondante. Le type de msgId est uint32. La gestion des valeurs de msgId relève de la responsabilité de l'**Hôte ECI**. Ces valeurs ne seront réémises qu'après le transfert du message de **Réponse**.

La **Réponse** contiendra un paramètre **msgResult** de type int32.

Ces paramètres implicites sont les premiers du champ payload d'un tampon de message. Le Tableau 9.3.2.3-1 présente la séquence de paramètres de ce champ pour chaque type de message du point de vue du **Client ECI** (le point de vue de l'**Hôte ECI** ne relève pas de l'interface **ECI**).

Tableau 9.3.2.3-1 – Types de messages et paramètres "cachés" (point de vue du Client)

Type de message	Paramètres implicites	Champ payload
Requête du client, C→H	*Aucun*	p_1, .. , p_n
Réponse de l'hôte, H→C	msgId, result	msgId, result, p_1, .. , p_n
Requête de l'hôte, H→C	msgId	msgId, p_1, .. , p_n
Réponse du client, C→H	msgId, result	msgId, result, p_1, .. , p_n

Les règles suivantes serviront à convertir les paramètres (structures, octets, matrices d'entiers courts, etc.) au format des données utiles du tampon de message dans l'espace mémoire du **Client ECI**:

- Les paramètres sont mappés dans la mémoire en commençant par leur adresse la plus faible, à l'exception des champs de données des matrices d'entiers à longueur variable.
- Le type de données 8 ou 16 bits est étendu à 32 bits à l'aide de l'extension adaptée au type concerné (signé ou non).
- Structures (hormis les champs de bits): tous les champs seront mappés dans l'ordre de leur définition. La taille du champ sera alignée (pour les entités 16 et 32 bits) sur le premier champ d'adresse la plus faible, le champ de bourrage précédant un champ suivant de plus grande taille. La structure est toujours rembourrée à la frontière 32 bits suivante. Les structures d'union seront rembourrées à la taille la plus élevée des alternatives.
- Les matrices d'octets (8 bits), courtes (16 bits) et de nombres entiers (32 bits) seront incluses dans le tampon du message (pas en tant que pointeurs vers la mémoire du **Client ECI**). Les matrices de longueur fixe utiliseront la notation <type>, <array_identifier>, '[' <constant> ']'. Elles seront mappées dans l'ordre de leur position dans la liste des paramètres. Les matrices de longueur variable utiliseront la notation <type>, <array_identifier>, '[' ']'. Toutes les matrices de longueur variable seront mappées sur deux champs de 32 bits. Le premier champ contient le décalage dans le tampon de message où se trouve le premier élément de la matrice. Le second champ contient la longueur de la matrice (en octets).
- Les entités de 64 bits seront stockées avec les 32 bits de poids le plus fort en premier (conformément aux conventions types de mappage des entités de 64 bits dans des machines 32 bits commençant par le bit de plus faible poids).
- La représentation naturelle (inconnue - définie par l'architecture sous-jacente du processeur central) de toutes les entités de 32 et 64 bits dans la mémoire commencera par le bit de plus faible poids.
- Tous les pointeurs vers des caractères imprimables (char *) utiliseront une représentation UTF-8 [ISO/CEI 21320] pour les "codes" réels sauf indication contraire explicite. La représentation des caractères pourra être de 1 à 4 octets (en fonction du code). Cette spécification ne définit pas les codes imprimables sur un **Équipement CPE** (dont la mise en œuvre peut varier selon les régions).

NOTE – L'**Hôte ECI** est responsable de l'interprétation de l'étiquette des messages en combinaison avec la version de l'API convenue avec le **Client ECI** pendant la phase de découverte. De la même manière, le **Client ECI** est responsable de l'interprétation de l'étiquette des messages en combinaison avec la version de l'API convenue avec l'**Hôte ECI** pendant la phase de découverte.

9.3.2.4 Convention de nommage des messages asynchrones

Convention relative aux noms de fonctions:

Tous les noms de fonctions commenceront par trois lettres correspondant au type de message. Le nom (<name>) de la fonction commencera par une majuscule. La définition de la convention de nommage des messages en fonction de leur type est donnée ci-dessous:

```
req<name>():    message de requête; res<name>():    message de réponse;
```

EXEMPLE 1: reqIpTcpSend().

Convention relative à la notation de paires de messages:

Les messages de **Requête** et de **Réponse** ainsi que les messages de demande et de réponse sont définis comme des paires. La notation applicable aux paires de messages est la suivante:

`<requestMessage>` → `<responseMessage>`

EXEMPLE 2: reqIpTcpSend(socket,buffer) → resIpTcpSend(socket).

Des signatures de fonction peuvent apparaître dans ces notations et d'autres sans saisir de paramètres par souci de concision.

Le Tableau 9.3.2.4-1 fournit quelques exemples de mappage pratique des noms de messages sur des fonctions possibles en langage C à l'aide d'une procédure, d'un événement de type javascript, d'approches de programmation du type abonnement/rappel ou de boucles d'envoi. La fonction **subscr** permet d'appeler une fonction lors de la réception d'un message doté d'une étiquette. Deux exemples sont donnés: le premier effectue une sélection en fonction de l'identificateur **msgId** et inclut une structure **cntxt** dans la fonction. Le second ne filtre pas en fonction de **msgId** et ne fournit pas de structure **cntxt** lors du rappel/de l'envoi.

Tableau 9.3.2.4-1 – Paramètres du champ payload par type de message avec les paramètres p_1, ... ,p_n

Message	Notation de type procédure	Rappel d'Evénement client Abonnement	Rappel client/notation d'envoi ou demande
Req, C→H	id = reqName([tag],p_1,..p_n)		
Res, H→C	res = resName([tag],id,p_1,..p_n)	subscr(tag,id,resName,cntxt) subscr(tag,resName)	resName(cntxt,res,p_1,..p_n) resName(id,p_1,..p_n)
Req, H→C	[tag =] reqName([id],p_1,..p_n)	subscr(tag,invName)	invName(id,p_1,..p_n)
Res, C→H	resName([tag],id,res,p_1..p_n)		

9.3.3 Messages synchrones

Comme les messages asynchrones, les messages synchrones adoptent une convention de notation faisant appel à des noms de fonctions. Les paramètres des messages synchrones ne seront pas mis en série pour s'adapter aux tampons des messages mais utiliseront les conventions générales du langage C relatives aux appels de fonctions et utiliseront la définition de l'interface binaire d'application de la machine virtuelle pour le mappage des procédures sur la mémoire de la machine virtuelle et sur l'état du registre. Cette approche permet de mapper directement les messages synchrones sur des fonctions normales en langage C dans le cadre de la bibliothèque des **Clients ECI**.

Il existe trois types prédéfinis: **get** pour lire une variable dans le domaine de l'**Hôte ECI**, **set** pour écrire une variable dans le domaine de l'**Hôte ECI** et **call**, une fonction générale avec renvoi d'un code d'erreur négatif ou d'une valeur de fonction non négative comme indiqué dans le Tableau 9.3.3-1.

Tableau 9.3.3-1 – Types de fonctions synchrones

Type	Application	Notation	Résultat	Sémantique
Get	Variable de l'hôte	getVariable((**i1**..**in**)	type de variable	Lit une variable indexée par les paramètres **i1..in** dans le domaine de l'**Hôte ECI** (pour le **Client ECI** concerné) (voir la NOTE).
Set	Variable de l'hôte	setVariable((**i1**..**in,** value)	void	Attribue une variable indexée par les paramètres **i1..in** dans le domaine de l'**Hôte ECI** (pour le **Client ECI** concerné) (voir la NOTE).
Call	Hôte	callFunc(p1..pn)	entier ou vide	Effectue un appel synchrone (général) à une fonction dans le domaine de l'**Hôte ECI**. La valeur renvoyée est du même type que la valeur du résultat pour les messages asynchrones: les valeurs négatives indiquent qu'une erreur est survenue. Certaines fonctions peuvent avoir un type vide (void) ne permettant pas de signaler les erreurs.
NOTE – Si la fonction Get est invoquée, l'**Hôte ECI** pourra être amené à effectuer des actions en sus du renvoi de l'objet demandé.				

Exemples de définition de messages synchrones:

```
   uint getClock();
   void setPwrWakeup (int timeout);
   void memcpy(char *p1, char *p2; int len) ;
```

Exemples d'utilisation:

```
   uint clock = getClock() ;       /* lecture de l'horloge */
   setPwrWakeup (1000);               /* réglage de la temporisation de réveil;
déclenchement des invocations */
   (void) memcpy(ptr1,ptr2,100*1000)  /* copie effective de la mémoire du
client */
```

9.3.4 Codes d'erreur relatifs aux Renvois

Le paramètre de code de Renvoi des **Réponses** et (le cas échéant) des **Appels** contiendra un unique nombre entier signé de 32 bits. Si la valeur renvoyée est nulle ou positive, l'exécution du code aura réussi. Une erreur renvoie une valeur négative. Les erreurs sont génériques (voir le Tableau 9.3.4-1) ou spécifiques à la **Requête** (voir les codes d'erreur propres à chaque **Requête**).

Tableau 9.3.4-1 – Codes d'erreur des messages de renvoi

Nom/Constante	Valeur	Description
	1..MaxInt	Succès de la **Requête**; valeur dépendant de la définition des messages.
ErrReqOkNoId	0	Succès de la **Requête**.
ErrReqApiErr	-1	API désignée par msgApiTag non prise en charge.
ErrReqCallErr	-2	Appel au sein d'une API désignée par msgApiTag non pris en charge.
ReqQueueErr	-3	Problème de mise en file d'attente du message, dépassement de la file d'attente du tampon de l'interface **ECI**.
ReqResource	-4	Problème de ressource pendant le traitement de la **Requête** (par exemple, problème de mémoire dû à un nombre excessif de messages).
RFU	-5..-15	Réservé à une utilisation future (types d'erreurs génériques).
ReqParam<N>Err	-16..-48	Paramètre **N** erroné = -Résultat-15.
Reserved for VM errors	-49..-64	Codes d'erreur réservés à des erreurs spécifiques à la machine virtuelle définis dans la Recommandation [UIT-T J.1013].
RFU	-65 .. -256	Réservé à une utilisation future.
Erreur spécifique à l'API.	-256 .. -511	Erreur spécifique à l'API définie par la table de codes d'erreur de l'API.
RFU	-512.. MinInt	Réservé à une utilisation future.

NOTE – En général, le **Client ECI** peut compter sur l'**Hôte ECI** pour prendre en charge un profil spécifique d'API (voir le § 9.2.5) et adopter une approche libérale des tampons de file d'attente des messages. Par conséquent, un traitement intelligent des erreurs n'est généralement pas requis. Habituellement le code d'erreur ne sert qu'aux scénarios de débogage des **Clients ECI**.

Les codes d'erreur spécifiques aux API ou le message ReqParamNErr ne peuvent pas faire partie d'un Renvoi. En revanche, l'erreur sera signalée dans la **Réponse**.

9.3.5 Canal authentifié sécurisé (SAC)

Les API du système de sécurité évoluée mettent à disposition des outils permettant d'établir un **Canal authentifié sécurisé (SAC)** entre un **Client ECI** et un autre dispositif approprié (voir le § 9.5.2). Le **Client ECI** qui a besoin d'une communication authentifiée et sécurisée avec un autre **Client ECI** ou un dispositif externe doit définir un mécanisme propriétaire capable d'utiliser les API disponibles, notamment celles du système de sécurité évoluée.

9.3.6 Vérification des messages par l'Hôte ECI

Afin d'éviter des conditions d'erreur ou des actions inappropriées du fait de **Requêtes** ou de **Réponses** inadaptées, les **Hôtes ECI** procéderont à la vérification complète de tous les messages reçus de la part d'un **Client ECI**. Les contrôles suivants seront effectués:

- Prise en charge de **msgApiTag**.
- Prise en charge de **msgCallId** dans l'espace de messages de l'API (dans sa version déterminée au moment de la découverte).
- Vérification de la correspondance entre les contraintes imposées aux données utiles, et notamment **msgLength**, et les règles de syntaxe du message, ainsi que du respect des portions de l'espace d'adresse du **Client ECI** allouées au tampon de message (pour les messages asynchrones) et à la mémoire de l'espace d'adresse du **Client ECI** que l'**Hôte ECI** devra lire ou dans lequel il devra écrire.
- Vérification de l'échec éventuel d'une **Précondition** propre aux messages (dans la mesure où la **Précondition** joue un rôle essentiel pour assurer l'intégrité de la **Requête** ou de la **Réponse**).
- Vérification de l'allocation de mémoire au **Client ECI** pour le pointeur ou la mémoire impliqués dans le message.

9.3.7 Traitement des messages par les Clients ECI

Toute mémoire allouée à l'envoi d'une **Requête** peut être réutilisée lors du renvoi, sauf indication contraire explicite (en général, s'il s'agit de messages volumineux qu'il est important d'éviter de copier). De la même manière, toute mémoire allouée à l'envoi d'une **Réponse** peut être réutilisée immédiatement après l'événement d'envoi.

Les **Clients ECI** n'attendront pas de **Réponse** des **Hôtes ECI** à chaque **Requête.**

Les **Clients ECI** pourront vérifier si la syntaxe d'une **Requête** ou d'une **Réponse** de l'**Hôte ECI** est correcte. Le **Client ECI** n'est pas obligé d'informer l'**Hôte ECI** en cas de **Requête** ou de **Réponse** mal formatée.

9.4 API relatives aux ressources générales de l'Hôte ECI

9.4.1 Liste des API définies dans le § 9.4

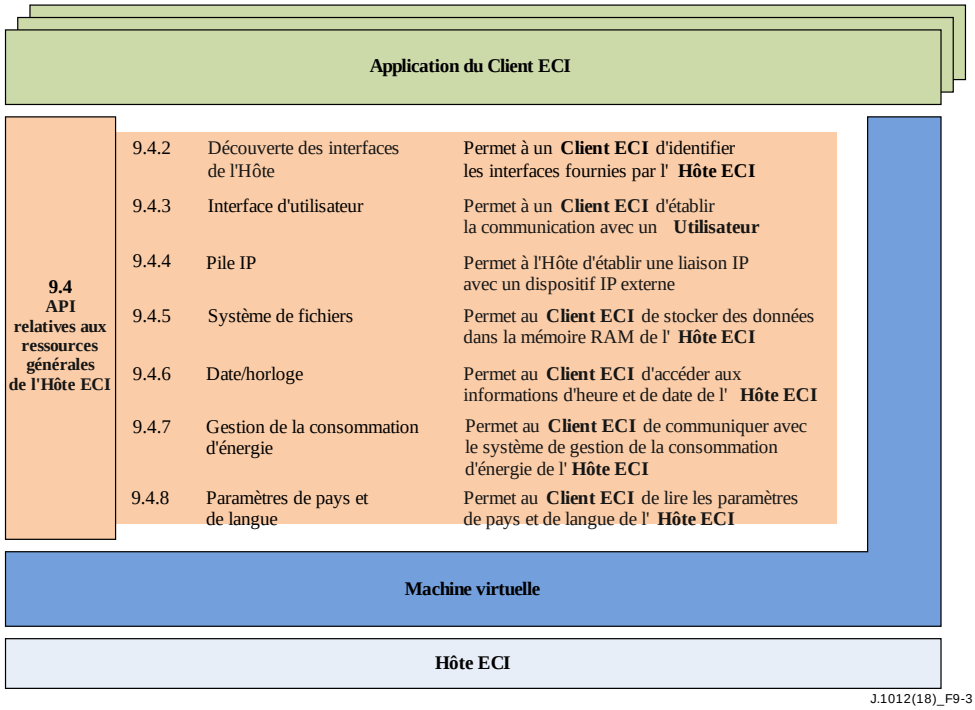


Figure 9.4.1-1 – Représentation des API définies dans le § 9.4

Tableau 9.4.1-1 – liste des API définies dans le § 9.4

Paragraphe	Nom de l'API	Description
9.4.2	Découverte des interfaces de l'Hôte	Permet à un **Client ECI** d'identifier les interfaces fournies par l'**Hôte ECI**.
9.4.3	Interface d'utilisateur	Permet à un **Client ECI** d'établir la communication avec un **Utilisateur**.
9.4.4	Pile IP	Permet à l'Hôte d'établir une liaison IP avec un dispositif IP externe.
9.4.5	Système de fichiers	Permet au **Client ECI** de stocker des données dans la mémoire RAM de l'**Hôte ECI**.
9.4.6	Date/Horloge	Permet au **Client ECI** d'accéder aux informations d'heure et de date de l'**Hôte ECI**.
9.4.7	Gestion de la consommation d'énergie	Permet au **Client ECI** de communiquer avec le système de gestion de la consommation d'énergie de l'**Hôte ECI**.
9.4.8	Paramètres de pays et de langue	Permet au **Client ECI** de lire les paramètres de pays et de langue de l'**Hôte ECI**.

Le Tableau 9.4.1-1 répertorie les API définies dans le § 9.4 et la Figure 9.4.1-1 illustre leur positionnement dans l'**Architecture ECI**.

Les tableaux utilisant la structure décrite dans le Tableau 9.4.1-2 fournissent une vue d'ensemble des messages de présentation des différentes API.

Tableau 9.4.1-2 – Structure du tableau résumant les fonctions des messages de chaque API

Message	Type	Sens	Étiquette	Description
Nom du message	Voir le Tableau 9.4.1-3	C→H ou H→C	Valeur de l'étiquette	Brève description de la fonction du message

La colonne Type du Tableau 9.4.1-2 indique le type de message, qui peut être synchrone ou asynchrone. Le Tableau 9.4.1-3 fournit des détails complémentaires. L'Annexe I répertorie l'ensemble des messages des API à la disposition des **Clients ECI**.

Tableau 9.4.1-3 – Valeurs possibles de la colonne Type

Catégorie de message	Notation dans la colonne Type	Commentaire
Message asynchrone	A	Types de messages possibles: voir le Tableau 9.3.2.3-1.
Message synchrone	A	Types de messages possibles: voir le Tableau 9.3.3-1.
	Set	
	Get	
	Call	

9.4.2 API d'accès à la ressource de découverte des interfaces de l'Hôte ECI

9.4.2.1 Introduction

Ce paragraphe définit l'API à la disposition des **Clients ECI** pour découvrir les API et leurs versions prises en charge par l'**Hôte ECI** et sélectionner la version la mieux adaptée à leurs sessions avec l'**Hôte ECI**. Le mécanisme de gestion des versions permet de les sélectionner API par API. La version sélectionnée sera utilisée jusqu'à l'initialisation suivante du **Client ECI** par l'**Hôte ECI**.

Les politiques relatives à la disponibilité des API sont traitées dans le § 9.2.5. Les API obligatoires sont définies dans le § 10.

Le **Client ECI** lancera la gestion des versions dès son initialisation: aucune API ne peut être utilisée sans version (mutuellement) approuvée par le Client et l'Hôte.

La version d'une API sera représentée par un nombre de 16 bits. La numérotation des versions d'API commence à 0x0000. L'attribution normale des nouvelles versions est incrémentielle (par pas de 1).

Le Tableau 9.4.2.1-1 répertorie les messages des API.

Tableau 9.4.2.1-1 – API de découverte des interfaces de l'Hôte ECI

Message	Type	Sens	Étiquette	Description
getApis	Get	C→H	0x0	Lit les API de l'Hôte disponibles.
getApiVersions	Get	C→H	0x1	Lit les versions disponibles d'une API de l'Hôte.
setApiVersion	Set	C→H	0x2	Ecrit la version de l'API de l'Hôte à utiliser.

9.4.2.2 Message getApis

C→H uint[] **getApis (**uint **maxNrApis**)

- Cette requête renvoie une table des bits de maxNrApis indiquant les API prises en charge par l'**Hôte ECI**.

Définition de la propriété:

- Existence d'une API de l'Hôte dotée d'une étiquette **a** où (**a**<**maxNrApis**) prend la forme **(*result*[a/32]>>(a%32))&0b1 == 0b1).**

Définition des paramètres:

maxNrApis: ushort	Numéro des API le plus élevé pour lequel renvoyer le résultat plus un.

9.4.2.3 Message getApiVersions()

C→H uint[] **getApiVersions (**uhort **api,** ushort **maxNrVersions**)

- Cette requête renvoie une table des bits de **maxNrVersions** indiquant les versions d'**api** prises en charge par l'**Hôte ECI**.

Définition de la propriété:

- Existence d'une version d'API dotée de l'étiquette **api** pour la version **v** où (**v**< **maxNrVersions)** prend la forme **(*result*[v/32]>>(v%32))&0b1 == 0b1).**

Définition des paramètres:

maxNrVersions: ushort	Numéro de version le plus élevé pour lequel renvoyer le résultat plus un.

9.4.2.4 Message setApiVersion()

C→H setApiVersion (ushort **api,** ushort **version**)

- Ce message définit la version de l'API à utiliser par le **Client ECI** et l'**Hôte ECI** pour **api** à **version**. Ne doit être appelé qu'une seule fois (les appels suivants sont sans effet).

Définition des paramètres:

api: ushort	Étiquette de l'API pour laquelle la version sera définie.
version: ushort	Numéro de version d'**api** à utiliser dans la session suivante entre le Client et l'Hôte.

Sémantique détaillée:

- Si **version** n'est pas une version existante d'API prise en charge par **api**, la version de l'API sera la première la plus élevée prise en charge, si elle est disponible. Sinon, il s'agira de la version la plus élevée de l'API.
- Les **Clients ECI** vérifieront la disponibilité d'une version d'API avant de l'initialiser.

NOTE – Sans vérification explicite, un comportement inattendu de l'API ou des conditions d'erreur risquent de se produire.

9.4.3 API d'accès à la ressource d'interfaces d'Utilisateur de l'Hôte ECI

9.4.3.1 Introduction

Ce paragraphe définit l'environnement des applications **ECI** permettant aux **Clients ECI** de mettre en place une interface d'interaction avec l'**Utilisateur**. Les applications **ECI** sont hébergées par des **Clients ECI** et exécutées sur un **Hôte ECI.** Elles utilisent un navigateur HTML présent dans les téléviseurs pour plusieurs plates-formes mises à disposition par des fournisseurs et des radiodiffuseurs.

La Figure 9.4.3.1-1 représente chaque entité de l'environnement des applications **ECI**. Le **Client ECI** ne contrôle pas l'application **ECI** qu'il a ouverte et ne communique pas directement avec elle. Il passe par un proxy fourni par l'**Hôte ECI**. Le proxy met en œuvre l'API définie au § 9.4.3.4, qui permet aux **Clients ECI** de lancer et d'arrêter des applications **ECI** ainsi que de communiquer avec celles en cours d'exécution, par exemple, pour traiter une entrée de l'**Utilisateur**. La communication entre l'application **ECI** et le **Client ECI** est gérée par le proxy, qui transcode la requête HTTP Get du navigateur soit en ressource issue du conteneur d'applications, soit en requête d'API reqUiClientQuery à destination du **Client ECI**, comme défini au § 9.4.3.4.8. Cette requête peut fournir au **Client ECI** l'entrée de l'**Utilisateur** et lui permettre d'y répondre avec un contenu dynamique. Le conteneur d'applications fournit les ressources statiques (plus larges) requises pour créer les écrans de l'interface d'utilisateur. Le **Client ECI** fournit une entrée personnalisée à l'écran de l'interface d'utilisateur et reçoit l'entrée de l'**Utilisateur**.

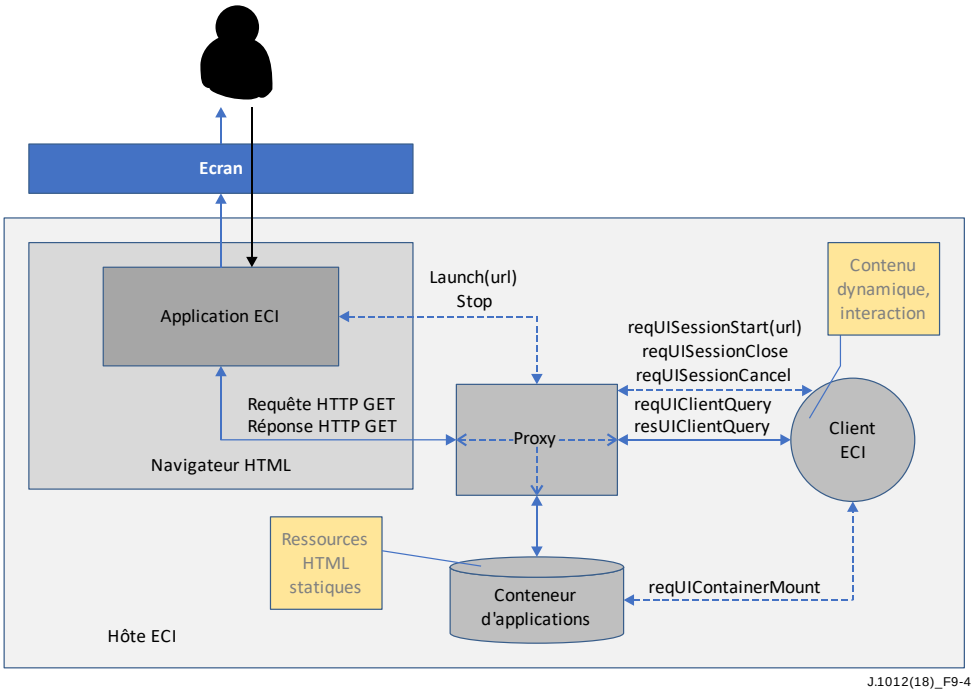


Figure 9.4.3.1-1 – Représentation de l'API d'interfaces d'Utilisateur

9.4.3.2 Environnement des interfaces d'Utilisateur

9.4.3.2.1 Profil du navigateur

L'**Hôte ECI** fournira un navigateur HTML mettant en œuvre le Web Standards TV Profile défini dans [CEI 62766-5-2] adapté aux contraintes et aux extensions définies dans la présente Recommandation. Ce profil est également adopté par le système HbbTV [b-HbbTV].

9.4.3.2.2 Contraintes

L'**Hôte ECI** refusera les requêtes HTTP envoyées à une ressource quelconque d'une session d'**Application ECI** n'émanant pas de la session en question.

Les adresses URL utilisées pour charger les ressources de l'**Application ECI** dans le navigateur résulteront de la concaténation d'une adresse URL de base propre à la session et d'une adresse URL relative constituant l'adresse du **Client ECI** ou du conteneur d'applications. Par exemple, si l'adresse URL de base de la session est:

```
http://localhost:3000/session-x/
```

et qu'une ressource du conteneur d'applications est:

```
main/pincode.html
```

l'adresse URL du navigateur est:

```
http://localhost:3000/session-x/main/pincode.html
```

Lorsque l'**Hôte ECI** gère des requêtes émanant du navigateur HTML, il doit déduire le type de contenu des ressources de l'**Application ECI** à partir de leur extension de nom de fichier et prendre au moins en charge:

- text/html - .html et .htm
- text/javascript - .js
- text/css - .css
- image/png - .png
- image/gif - .gif
- image/jpeg - .jpg et .jpeg

9.4.3.2.3 Capacités du navigateur

9.4.3.2.3.1 Type d'affichage

La fenêtre du navigateur sera au format plein écran. Ses dimensions seront de 1 280 x 720 pixels minimum. La conception des applications **ECI** devra leur permettre de s'adapter à des dimensions supérieures.

Le volet graphique affichant les applications **ECI** se trouvera derrière celui des applications du terminal et devant tous les autres, y compris ceux relatifs aux applications de vidéo, de sous-titrages et de radiodiffusion.

Le volet des applications **ECI** recouvre entièrement tous les volets graphiques, hormis celui du terminal. L'arrière-plan de la fenêtre du navigateur doit être transparent: si une zone n'est pas couverte par un élément HTML de l'application **ECI**, les volets graphiques situés en dessous (dont l'un contient en général la diffusion vidéo) doivent être visibles. Si la couleur d'arrière-plan des propriétés CSS de l'élément corps (body) est définie comme transparente, la fenêtre d'arrière-plan du navigateur sera transparente.

Lorsque le terminal doit temporairement recouvrir l'application **ECI** – par exemple, pour afficher le menu système ou une bannière d'information sur les chaînes suite à une action de l'**Utilisateur** –, l'application **ECI** ne peut plus recevoir d'entrées. Dans ce cas, un événement de brouillage ciblera l'objet Fenêtre.

Lorsque le terminal fermera son interface d'utilisateur alors que l'application **ECI** est toujours en cours d'exécution, celle-ci pourra à nouveau recevoir des entrées. Dans ce cas, un événement d'activation ciblera l'objet Fenêtre. Le navigateur prendra en charge le format de couleurs RGBA32.

9.4.3.2.3.2 Texte et polices de caractères

Le navigateur comprendra une police proportionnelle intégrée. Les applications **ECI** pourront utiliser les noms génériques de familles de caractères "sans serif" ou "par défaut" pour sélectionner la police intégrée. Le jeu de caractères pris en charge par la police intégrée doit être adapté à la région où le dispositif est déployé. Alternativement, les applications **ECI** pourront utiliser les polices web CSS3 et les jeux de caractères définis dans [CEI 62766-5-2]. Le navigateur prendra en charge au moins une police web téléchargeable par application **ECI**.

Le navigateur prendra en charge le codage UTF-8 pour toutes les ressources texte d'une application **ECI**, c'est-à-dire les documents HTML, les scripts et les feuilles de style.

9.4.3.2.3.3 Formats graphiques

Le navigateur prendra en charge les formats graphiques suivants: GIF [W3C GIF V89a], JPEG [UIT-T T.871] et PNG [W3C PNG].

9.4.3.2.3.4 Entrées de l'Utilisateur

Le navigateur prendra en charge les entrées de l'**Utilisateur** par télécommande à l'aide de DOM3 KeyboardEvents. Lorsqu'une application **ECI** sera en cours d'exécution et en mesure de recevoir des entrées, l'**Hôte ECI** autorisera l'**Utilisateur** à lancer les événements suivants:

- Touches numériques: 0-9.
- Touches de curseur: gauche, droite, haut, bas, entrée et retour navigateur.

La prise en charge des attributs hérités keyCode et charCode n'est pas obligatoire.

9.4.3.2.3.5 Persistance

Le navigateur prendra en charge le stockage des sessions de l'API WebStorage et des cookies de session. Les **Clients ECI** conserveront les informations des différentes sessions dans leur mémoire interne.

9.4.3.2.3.6 Accès des Applications ECI aux ressources HTML statiques

Le proxy recevant les requêtes HTTP de l'**Application ECI** mappera l'adresse URL relative (c'est-à-dire l'extension de l'adresse URL de base de la session) sur un chemin relatif dans le conteneur d'applications monté par le **Client ECI**. Le mappage entre l'adresse URL relative et le fichier est direct: les paramètres directoryname1/directoryname2/.. / directorynameN/filename de l'adresse URL relative sont mappés sur le nom de fichier (filename) dans le répertoire directorynameN contenu dans ... contenu dans le répertoire directoryname2 contenu dans le répertoire directoryname1.

La structure et les fichiers des répertoires du conteneur d'applications respecteront les contraintes suivantes:

- Tous les noms de fichiers et les répertoires seront constitués de caractères alphanumériques et des caractères '.' (point) et '_' (trait de soulignement) (40 maximum).

D'autres ressources ou exigences de performance relatives au conteneur d'applications sont proposées dans le document [b-UIT-T J Suppl. 7].

9.4.3.2.3.7 Communication entre le Client ECI et les Applications ECI

Le navigateur prend en charge l'API XmlHttpRequest comme l'exige le § 9.4.3.2.1 de la présente Recommandation. La communication entre les applications **ECI** et les **Clients ECI** est acheminée via le proxy de l'**Hôte ECI**. L'application **ECI** peut effectuer une requête HTTP Get à l'aide de l'API XMLHttpRequest définie dans ce paragraphe. L'adresse URL de la requête HTTP sera construite à partir de l'adresse URL de base de la session de l'**Application ECI** définie au § 9.4.3.2.2 et de l'adresse URL relative '/client'. Les éventuels paramètres feront partie de la chaîne d'interrogation

sous la forme de paires clé-valeur. Les clés et les valeurs ne comporteront que des caractères ASCII. La longueur des clés sera limitée à 31 caractères et celle des valeurs à 255 caractères au maximum.

EXEMPLE: http://localhost:3000/ session-20170303-163100-01/client?id=e4f0&p2=v2.

A réception de la requête HTTP, le proxy de l'**Hôte ECI** enverra un message reqUiClientQuery au **Client ECI** de l'application **ECI** comme indiqué au § 9.4.3.4.5. La clé de requête analysée constituera les paires de valeurs de clés. La réponse du **Client ECI** à l'Hôte inclura les paramètres suivants:

- type: chaîne respectant les types de médias définis par les normes pertinentes, et documentés dans la base de données IANA [b-IANA], tels que application/json définis par [b-IETF RFC 8259].
- status code: nombre entier utilisé dans la réponse de la requête Get 200 correspond au succès de la requête.
- body: chaîne de 64 ko maximum.

À partir de ces valeurs, l'**Hôte ECI** construira la réponse de la requête HTTP Get à l'intention du navigateur en définissant l'en-tête Content-Type sur le paramètre 'type', le statut HTTP sur la valeur d'erreur et le corps de la réponse sur la valeur du paramètre 'body'.

La communication avec les applications HTML ne provenant pas du **Client ECI** ne relève pas de cette version de la présente Recommandation.

9.4.3.3 Cycle de vie des applications

9.4.3.3.1 Lancement des applications ECI

L'écran du téléviseur est une ressource partagée où coexistent des applications de terminal, de radiodiffusion, de l'**Opérateur** et de tiers. Cette version de la présente Recommandation définit l'environnement des applications d'interfaces d'**Utilisateur** de base requises pour le fonctionnement de modules **ECI**, par exemple, saisie d'un code PIN, informations d'abonnement, etc.

Les requêtes de lancement émises par les **Clients ECI** sur les **Hôtes ECI** seront limitées aux cas suivants:

- L'**Hôte ECI** est sur le point de démarrer la présentation de médias (par exemple, après le passage à une chaîne de radiodiffusion) que le **Client ECI** est en train de traiter.
- L'**Hôte ECI** est en train de présenter des médias traités par le **Client ECI**.
- L'**Hôte ECI** a demandé au **Client ECI** d'afficher son **Menu des applications**.
- Le **Client ECI** indique son intention de lancer un flux autre qu'un contenu en rapport avec l'**Application ECI**, et l'**Hôte ECI** s'assure que l'**Utilisateur** est l'auteur de la requête de dialogue ou que celle-ci ne provoque pas de conflit avec le contenu affiché à l'écran, c'est-à-dire qu'il n'y a ni suppression/masquage d'un contenu de tiers que l'**Utilisateur** souhaite visionner ni superposition d'écran.

Dans les cas ci-dessus, une requête de lancement visant à effectuer une interaction d'authentification parentale déléguée telle que définie au § 9.8.2.11 avec l'**Utilisateur** est considérée comme une requête de lancement à l'initiative du **Client ECI** émetteur de la requête d'authentification parentale d'origine définie au § 9.8.2.10.

On entend par **Conflit d'écran** une situation où le **Client ECI** demande à l'**Hôte ECI** de lancer une **Application ECI** (ouvrir une session d'interface d'utilisateur) alors que les conditions de lancement ci-dessus ne sont pas satisfaites.

Si l'**Hôte ECI** est en mesure d'exécuter des applications interactives, il pourra lancer au moins une **Application ECI** pendant l'exécution d'un contenu interactif lié au média présenté à l'écran. Cette **Application ECI** présentera un rapport direct avec le média affiché. Le lancement de l'**Application ECI** ne mettra pas fin au contenu interactif présenté à l'écran et ce contenu pourra reprendre son interaction avec l'**Utilisateur** lorsque l'exécution de l'**Application ECI** cessera.

L'**Hôte ECI** informera l'**Utilisateur** qu'un **Client ECI** souhaite lancer un flux autre que du contenu en rapport avec l'**Application ECI** ou autorisera le **Client ECI** à lancer régulièrement cette **Application** sans **Conflit d'écran**. Pour ce faire, il sera possible, par exemple, de lancer ces **Applications ECI** au moment de la mise sous tension ou du passage en mode veille ou bien de recourir à une action de l'**Utilisateur** en réponse à une icône d'alerte dans une bannière ou à un écran de menu de l'**Hôte ECI** s'affichant normalement. Les **Clients ECI** ne doivent pas avoir la possibilité de lancer fréquemment ce type d'**Applications ECI** et doivent limiter leur usage à des questions importantes en vue de garantir un fonctionnement continu.

L'**Application ECI** lancée par le **Client ECI** sera chargée dans des contextes de navigation non accessibles à partir du navigateur d'une application de radiodiffusion ou de tiers.

La fenêtre du navigateur sera visible en une seconde et devra avoir intégralement chargé l'**Application ECI**.

Éventuellement, les versions futures de la présente Recommandation prescriront des modèles de cycle de vie et des mécanismes de résolution des conflits étendus et autoriseront la communication avec des applications HTML lancées depuis l'extérieur.

9.4.3.3.2 Fermeture des Applications ECI

Pour arrêter une **Application ECI**, le **Client ECI** envoie un message reqUISessionStop à l'**Hôte ECI**. La requête comprend un message uiSessionId renvoyé par l'**Hôte ECI** dans la réponse resUISessionOpen. L'application **ECI** s'arrête. Les modalités de sa fermeture dépendent de la mise en oeuvre: arrêt du navigateur ou réduction de sa fenêtre, notamment. Dans tous les cas, il ne sera plus possible d'entrer des informations dans l'application **ECI** et le navigateur ne lui enverra plus de messages KeyboardEvents.

L'application **ECI** s'arrêtera également si une action de l'**Utilisateur** (par exemple, appuyer sur P+/P-) implique que le terminal passe à un état interdisant le lancement de cette application. L'**Hôte ECI** enverra un message reqUiSessionCancel au **Client ECI**.

9.4.3.4 API relatives à la communication avec l'Utilisateur

9.4.3.4.1 Liste des messages des API de communication avec l'Utilisateur

L'API d'interface d'**Utilisateur** permet au **Client ECI** de monter un fichier conteneur d'applications d'interfaces d'utilisateur téléchargé qui fournira l'essentiel des ressources HTML statiques requises pour générer l'interface d'**Utilisateur**. Le proxy résout automatiquement toutes les requêtes HTTP envoyées par le navigateur au fichier conteneur d'applications ne concernant pas un client.

L'**Hôte ECI** peut suggérer au **Client ECI** de démarrer une application soit en réponse à une demande d'accès au **Menu des applications** du **Client ECI** émanant de l'**Utilisateur**, soit en indiquant au **Client ECI**, à l'aide du message reqUiSessionCommence, l'absence d'un conflit susceptible de l'empêcher de présenter à l'**Utilisateur** une **Application ECI** non associée un **Pointeur de média**. Le **Client ECI** peut indiquer son souhait de lancer un dialogue non associé à un **Pointeur de média** de ce type à l'aide du message SetUiClientAttention. Cette possibilité abaisse le niveau de priorité de la communication entre le **Client ECI** et l'**Utilisateur** en l'absence de **Conflit d'écran**.

Le **Client ECI** ouvre toutes les sessions d'interface d'**Utilisateur** à l'aide du message reqUiSessionOpen. L'adresse URL relative donnant lieu au premier écran d'interface d'utilisateur est fournie sous forme de paramètre. Le **Client ECI** et l'**Hôte ECI** peuvent mettre fin à la session de l'interface d'**Utilisateur** avec les messages reqUiSessionClose et reqUiSessionCancel respectivement.

Le message ReqUiClientQuery permet à l'**Application ECI** du navigateur d'envoyer via le proxy des requêtes dotées de paramètres au **Client ECI**, qui peut alors y répondre avec des données destinées à l'application HTML. Ce mode de communication permet à l'**Application ECI** de présenter des données spécifiques au **Client ECI** et de fournir à celui-ci des entrées de l'**Utilisateur** de la même manière qu'une application HTML communiquant avec un serveur HTTP dynamique.

Le Tableau 9.4.3.4.1-1 répertorie les API définies dans le présent paragraphe.

Tableau 9.4.3.4.1-1 – Messages des API d'interfaces d'Utilisateur

Message	Type	Sens	Étiquette	Description
reqUiContainerMount	A	C→H	0x0	Monte un conteneur d'applications d'interfaces d'utilisateur à l'aide de ressources HTML afin de prendre en charge des sessions d'interface d'utilisateur.
setUiClientAttention	S	C→H	0x1	Le **Client ECI** demande à démarrer une session d'interface d'utilisateur sans association à un **Pointeur de média.**
reqUiSessionCommence	A	H→C	0x2	L'**Hôte ECI** suggère au **Client ECI** d'ouvrir une session d'interface d'utilisateur.
reqUiSessionOpen	A	C→H	0x3	Le **Client ECI** demande l'ouverture d'une session d'interface d'**Utilisateur** avec l'**Utilisateur** et affiche un contenu.
reqUiSessionClose	A	C→H	0x4	Le **Client ECI** met fin à une session d'interface d'**Utilisateur**.
reqUiSessionCancel	A	H→C	0x5	L'**Hôte ECI** annule une session d'interface d'**Utilisateur**.
reqUiClientQuery	A	H→C	0x6	Le **Client ECI** reçoit une requête de l'application HTML dans le navigateur et fournit une réponse (dynamique).

9.4.3.4.2 Message reqUiContainerMount

C→H reqUiContainerMount(fileName **filename**, PubKey **pk**) →
H→C resUiContainerMount (uint **indexFileLen**, uchar **indexFile**)

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** de désigner un fichier comme conteneur d'applications contenant les ressources HTML de son **Application ECI**. Si l'opération réussit, elle renvoie le contenu du fichier "EciIndex.txt" dans le répertoire principal du conteneur d'applications.

Définitions des paramètres de la requête:

filename: fileName	Nom du fichier du système de fichiers du client ECI qui constituera le conteneur d'applications désigné.
pk: PubKey	Clé publique permettant de vérifier la signature du conteneur d'applications.

Définitions des paramètres de la réponse:

indexFileLen: uint	Longueur du fichier d'index.
indexFile: uchar	Contenu du fichier d'index.

Sémantique détaillée:

- Les crochets [et] encadrant un texte comme ci-dessous indiquent la démarcation des champs et des structures dans les conteneurs de fichiers ZIP.
- La signature de vérification du fichier conteneur figure dans le champ [.ZIP file comment] de la structure [end of central directory record structure] (voir la Spécification du format de fichiers Zip, version 6.3.3 de PKWARE® Inc. mentionnée dans [ISO/CEI 21320]).
- Le champ [.ZIP Comment Field] est défini *à la fin* de la chaîne suivante composée de tous les caractères ASCII: ECI_SIGNATURE=" suivi de la valeur de la structure ECI_Data_Signature définie dans le Tableau 5.6-1 codée sous forme de chaîne hexadécimale avec des majuscules, suivie de ".

EXEMPLE:

ECI_SIGNATURE="01000000FCB1F60456719035FCB1F60456719035FCB1F60456719035FC
B1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719
035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60
456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FC
B1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719
035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60
456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FCB1F60456719035FC
B1F60456719035"

La longueur de la chaîne de signature de données codée d'un champ ECI_Data_Signature de type -1 est de 520 caractères.

- L'**Hôte ECI** vérifiera la signature calculée à partir du fichier conteneur jusqu'au champ [.Zip comment field] de la structure [end of central directory record] et définira le champ [.Zip comment field] sur 0x0000 à l'aide du paramètre de clé publique pk et du processus de calcul des signatures décrit au § 5.6.
- Le fichier d'index est le fichier nommé "EciIndex.txt" dans le répertoire principal du fichier conteneur.
- Le **Client ECI** montera un conteneur d'applications d'interface d'utilisateur valable si la conduite des sessions d'interface d'utilisateur l'exige.
- Le **Client ECI** sera en mesure d'afficher un message d'avertissement de base à l'intention de l'**Utilisateur** en cas d'échec du chargement et du montage du conteneur d'applications de l'interface d'utilisateur.

Remarques relatives aux applications:

- Les clients peuvent charger des fichiers conteneurs d'applications dans leur système de fichiers à partir d'un serveur en ligne utilisant l'API HTTP(S) (voir le § 9.4.4.6) ou d'un flux de transport de radiodiffusion utilisant l'API de carrousels de données.
- Le fichier "EciIndex.txt" peut contenir des informations de version pour l'interface d'utilisateur, vérifiées par la signature de clé publique.

Les codes d'erreur relatifs au message reqUiContainerMount sont définis dans le Tableau 9.4.3.4.2-1.

Tableau 9.4.3.4.2-1 – Codes d'erreur relatifs au message reqUiContainerMount

Nom	Description
ErrUiContainerFileNot	Voir le Tableau 9.4.3.4.9-1
ErrUiContainerNot	
ErrUiContainerSignature	
ErrUiContainerIndexTxtNot	

9.4.3.4.3 Message setUiClientAttention

C→H setUiClientAttention(uint **clientAttention**)

- Ce message indique que le **Client ECI** souhaite ouvrir une session d'interface d'utilisateur avec l'**Utilisateur** non associée à un **Pointeur de média** (type de session d'interface d'utilisateur égal à EciUiSessionDiaReq, voir le § 9.4.3.4.4).

Définition de la propriété:

clientAttention: uint	Les valeurs définies sont les suivantes: 0x0: il n'est pas souhaitable d'attirer l'attention de l'**Utilisateur**. 0x1: il est souhaitable d'attirer l'attention de l'**Utilisateur**. Toutes les autres valeurs sont réservées.

Postconditions:

- Si clientAttention=0x0, l'**Hôte ECI** n'émettra pas de messages reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq).
- Si clientAttention=0x1, l'**Hôte ECI** émettra un message reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq) à condition qu'aucun message de ce type ne soit en instance.

9.4.3.4.4 Message reqUiSessionCommence

H→C reqUiSessionCommence (uint **uiSessionType**) →
C→H resUiSessionCommence ()

- Ce message permet à l'**Hôte ECI** de suggérer au **Client ECI** d'ouvrir une session d'interface d'utilisateur d'un type spécifique.

Définitions des paramètres de la requête:

uiSessionType: uint	Nom du fichier figurant dans le système de fichiers du **Client ECI** qui constituera le conteneur d'applications désigné. Les valeurs sont définies dans le Tableau 9.4.3.4.4-1. Seules les valeurs EciUiSessionAppMenu et EciUiSessionDiaReq sont autorisées.

Tableau 9.4.3.4.4-1 – Types de sessions d'interfaces d'Utilisateur ECI

Nom	Valeur	Description
EciUiSessionDiaReq	0x00	Le **Client ECI** a demandé une session d'interface d'utilisateur avec l'**Utilisateur** via le message setUiClientAttention (non associée à un **Pointeur de média** spécifique) et l'**Hôte ECI** peut accéder à la requête d'ouverture que lui a envoyée le **Client ECI** via le message reqUISessionOpen.
EciUiSessionAppMenu	0x01	**Menu d'applications** du **Client ECI**. Permet à l'**Utilisateur** d'accéder à son initiative à l'ensemble des paramètres, des informations et des fonctions qu'il est en mesure de lancer.
EciUiSessionMh	0x02	Session d'interface d'utilisateur demandée par le **Client ECI**, associée à des opérations en rapport avec un **Pointeur de média.**
EciUiSessionParAuthDel	0x03	Session d'interface d'utilisateur demandée par le **Client ECI** dans le but de conduire un dialogue d'authentification parentale déléguée afin de traiter du contenu sur un **Pointeur de média**.
RFU	Autre	Réservé à une utilisation future.

NOTE – L'ordre de priorité des valeurs définies dans le Tableau 9.4.3.4.4-1 est fourni à titre de suggestion. Il peut servir d'indication pour résoudre les conflits de travail avec l'interface d'utilisateur dans la conception de l'**Hôte ECI**.

Sémantique détaillée:

- Le **Client ECI** pourra présenter un **Menu d'applications**. Au minimum, ce dernier devra permettre à l'**Utilisateur** d'inspecter la version du **Client ECI**, une référence à l'**Opération de plate-forme** et l'état opérationnel du **Client ECI**.

Préconditions de la requête:

- Aucun message reqUiSessionCommence antérieur en instance relatif à une session d'interface d'utilisateur n'aura été envoyé au **Client ECI**.

Postconditions de la réponse:

- Le **Client ECI** émettra un message reqUiSessionOpen assorti du type de session d'interface d'utilisateur correspondant, sinon une erreur sera signalée.

Les codes d'erreur relatifs au message reqUiSessionCommence sont définis dans le Tableau 9.4.3.4.4-2.

Tableau 9.4.3.4.4-2 – Codes d'erreur du message reqUiClientSessionCommence

Nom	Description
ErrUiResourceError	Voir le Tableau 9.4.3.4.9-1.
ErrUiClientError	

9.4.3.4.5 Message reqUiSessionOpen

C→H reqUiSessionOpen(uint **uiSessionType,** ushort **mH,** uint **relUrlLen,** char **relUrl[]) → H→C resUiSessionOpen(**ushort **uiSessionId)**

- Ce message permet au **Client ECI** de demander une nouvelle session d'interface d'utilisateur à l'**Hôte ECI**.

Définitions des paramètres de la requête:

uiSessionType: uint	Type de session d'interface d'utilisateur défini dans le Tableau 9.4.3.4.4-1. Le paramètre mH n'aura de pertinence que si la valeur est EciUiSessionMh ou EciUiSessionParAuthDel.
mH: ushort	Pointeur de média de la session de traitement des contenus à laquelle l'interface homme-machine est associée.
relUrlLen: uint	Longueur de relUrl en octets.
relUrl: char[]	Adresse URL relative, terminée par un caractère nul. Annexée à l'adresse URL de base de la session, elle formera l'adresse URL du navigateur qui démarrera la session d'interface d'utilisateur. Voir le § 9.4.3.2.2.

Définitions des paramètres de la réponse:

uiSessionId: ushort	Identificateur de la nouvelle session d'interface d'utilisateur.

Sémantique détaillée:

- Les **Clients ECI** seront en mesure de gérer simultanément plusieurs sessions d'interface d'utilisateur. Cependant, ils ne seront obligés de prendre en charge qu'une seule session d'interface d'utilisateur simultanée du type EciUiSessionAppMenu ou EciUiSessionAppMenu et au maximum une seule session d'interface d'utilisateur de type EciUiSessionMh par **Pointeur de média** ouvert.
- Les **Clients ECI** pourront ouvrir simultanément des sessions d'interface d'utilisateur de type EciUiSessionMh.
- Les **Clients ECI** qui prennent en charge l'API de délégation d'authentification parentale seront en mesure d'ouvrir simultanément des sessions d'interface d'utilisateur de type EciUiSessionParAuthDel. Celles-ci pourront s'exécuter parallèlement à d'autres sessions d'interface d'utilisateur des **Clients ECI**.
- Un **Hôte ECI** pourra prendre en charge une ou plusieurs sessions d'interface d'utilisateur simultanées, conformément aux modes des applications des **Équipements CPE**.

Préconditions de la requête:

1) Si la valeur de la requête uiSessionType est EciUiSessionAppMenu ou EciUiSessionDiaReq, ce message aura été précédé d'un message reqUiClientCommence doté du même paramètre uiSessionType.
2) Si la valeur de uiSessionType est EciUiSessionParAuthDel, ce message aura été précédé du message reqParAuthDel pour le Pointeur de média mH envoyé par l'**Hôte ECI** au **Client ECI**.
3) Si la valeur de la requête uiSessionType est EciUiSessionMh, Mh correspondra à une session de Pointeur de média ouverte.

Préconditions de la réponse:

1) Si la valeur de la requête uiSessionType est EciUiSessionAppMenu, EciUiSessionDiaReq ou EciUiSessionParAuthDel, l'**Hôte ECI** n'acceptera la requête de session d'interface d'utilisateur que dans les cas suivants: il l'a émise précédemment, la cause de la requête n'a pas été amendée et son état n'est pas susceptible de provoquer un **Conflit d'écran**.

2) Si la valeur de la requête uiSessionType est EciUiSessionMh, l'**Hôte ECI** accèdera à la requête de session d'interface d'utilisateur s'il est en mesure d'établir une interaction significative avec l'**Utilisateur** sans générer de conflit de priorité d'écran.

3) Les **Hôtes ECI** ne refuseront pas la demande de seconde session émanant d'un **Client ECI** si cette session possède un type uiSessionType égal à EciUiSessionParAuthDel. L'**Hôte ECI** a le droit d'annuler la première session.

Remarques relatives aux applications:

1) L'**Hôte ECI** refusera les sessions de **Pointeur de média** utilisées pour l'enregistrement s'il lui est impossible de dialoguer avec l'**Utilisateur** sans provoquer un **Conflit d'écran** ou si aucun écran n'est actif.

2) Il est recommandé que les applications des **Hôtes ECI** puissent ouvrir des sessions d'interface d'utilisateur aux fins d'authentification parentale, par exemple, lors de la programmation d'enregistrements donnant lieu à l'envoi du message reqParAuthCid de l'API d'authentification parentale (voir le § 9.8.2.10).

3) Les **Hôtes ECI** peuvent annuler une session d'interface d'utilisateur avec un **Client ECI** afin d'autoriser une nouvelle session dont le type uiSessionType est égal à EciUiSessionParAuthDel ou EciUiSessionMh.

Les codes d'erreur relatifs au message reqUiSessionOpen sont définis dans le Tableau 9.4.3.4.5-1.

Tableau 9.4.3.4.5-1 – Codes d'erreur du message reqUiClientSessionStart

Nom	Description
ErrUiScreenConflict	Voir le Tableau 9.4.3.4.9-1
ErrUiNoScreen	

9.4.3.4.6 Message reqUiSessionClose

C→H reqUiSessionClose(ushort **uiSessionId) →**
H→C resUiSessionClose(ushort **uiSessionId)**

- Ce message permet au **Client ECI** de fermer une session d'interface d'utilisateur existante.

Définitions des paramètres de la requête:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur à fermer.

Définitions des paramètres de la réponse:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur fermée.

Préconditions de la requête:

1) Une session d'interface d'utilisateur associée à uiSessionId sera ouverte.

2) Aucun autre message faisant référence à uiSessionId ne sera envoyé à l'**Hôte ECI**.

Préconditions de la réponse:

1) Aucun autre message faisant référence à uiSessionId ne sera envoyé au **Client ECI**.

9.4.3.4.7 Message reqUiSessionCancel

H→C reqUiSessionCancel (ushort **uiSessionId,** uint **reason) →**
C→H resUiSessionCancel (ushort **uiSessionId)**

- Ce message permet à l'**Hôte ECI** de fermer une session d'interface d'utilisateur existante avec le **Client ECI**. L'**Hôte ECI** utilise ledit message lorsque les conditions d'affichage d'une **Application ECI** ne sont plus satisfaites, par exemple si le passage de l'**Utilisateur** à une autre chaîne appartenant à un **Client ECI** différent provoque un **Conflit d'écran**.

Définitions des paramètres de la requête:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur à annuler.
reason: uint	Motif de l'annulation de la session. Les valeurs sont définies dans le Tableau 9.4.3.4.9-1.

Définitions des paramètres de la réponse:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur annulée.

Préconditions de la requête:

1) La session d'interface d'utilisateur associée à uiSessionId sera ouverte.
2) Aucun autre message faisant référence à uiSessionId ne sera envoyé.

Préconditions de la réponse:

1) Aucun autre message faisant référence à uiSessionId ne sera envoyé.

9.4.3.4.8 Message reqUiClientQuery

H→C reqUIClientQuery(ushort **uiSessionId,** uint **queryLen,** KeyValPair **query[]) →**
C→H resUIClientQuery(ushort **uiSessionId,** uint **statusCode,** uint **typeLen,** char **type[],** uint **bodyLen,** uchar **body[])**

- Ce message envoie une requête HTTP de l'**Application ECI** s'exécutant dans le navigateur de l'**Hôte ECI** comme décrit au § 9.4.3.2.3.7 et permet au **Client ECI** d'envoyer une réponse HTTP à l'**Application ECI**.

Définitions des paramètres de la requête:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur émettrice de la requête.
queryLen: uint	Longueur du paramètre de requête en octets.
query[]: KeyValPair	Contient les paires valeur-clé des paramètres de demande de la requête HTTP émise par le navigateur.

Définitions du type KeyValPair

```
#define MaxKeyLen 32
#define MaxValLen 256

typedef struct KeyValPair {
   char key[MaxKeyLen];  /* Clé de la paire valeur-clé, terminée par un
zéro*/
   char val[MaxValLen];  /* Valeur de la paire valeur-clé, terminée par un
zéro*/
KeyValPair
```

Définitions des paramètres de la réponse:

uiSessionId: ushort	Identificateur de la session d'interface d'utilisateur.
statusCode: uint	Code de statut HTTP tel que défini dans [IETF RFC 7231].
typeLen: uint	Longueur du paramètre 'type' en octets.
type[]: char	Type de réponse sous la forme d'une chaîne de caractères ASCII se terminant par un zéro.
bodyLen: uint	Longueur du paramètre 'body' en octets.
body[]: uchar	Message de réponse HTTP.

Préconditions de la requête:

1) La session identifiée par **uiSessionId** est ouverte.

Sémantique détaillée:

- Si l'**Application ECI** envoie une chaîne de requête mal formatée, l'**Hôte ECI** peut renvoyer un code de statut HTTP 400 et ne pas envoyer de requête au **Client ECI**.
- La relation entre les paramètres du message et la requête HTTP ainsi que la réponse HTTP du navigateur sont définies au § 9.4.3.2.3.7.

9.4.3.4.9 Codes d'erreur de l'API de communication avec l'Utilisateur

Les codes d'erreur relatifs à la communication avec l'interface d'**Utilisateur** sont répertoriés dans le Tableau 9.4.3.4.9-1.

Tableau 9.4.3.4.9-1 – Codes d'erreur relatifs à l'API de communication avec l'Utilisateur

Nom	Valeur	Description
ErrUiContainerFileNot	-256	Conteneur d'applications d'interface d'utilisateur introuvable.
ErrUiContainerNot	-257	Le fichier n'est pas un fichier de conteneur d'applications d'interface d'utilisateur valable.
ErrUiContainerSignature	-258	Echec du contrôle de la signature du fichier de conteneur d'applications.
ErrUiContainerIndexTxtNot	-259	Pas de fichier "EciIndex.txt" dans le répertoire supérieur du conteneur d'applications.
ErrUiResourceError	-260	Le **Client ECI** ne peut pas monter la ressource du conteneur d'applications de l'interface d'utilisateur.
ErrUiClientError	-261	L'état opérationnel du **Client ECI** ne lui permet pas de présenter d'interface d'utilisateur.
ErrUiDiaNoMore	-262	La requête de dialogue émanant du **Client ECI** n'est plus valable.
ErrUiScreenConflict	-263	L'**Hôte ECI** présente un **Conflit d'écran** et ne peut ni ouvrir ni mener une session.
ErrUiNoScreen	-264	L'**Hôte ECI** n'a pas ou plus accès à un écran pour présenter la session d'interface d'utilisateur.
RFU	Autre	Réservé à une utilisation future.

9.4.4 API d'accès à la ressource de pile IP de l'Hôte ECI

9.4.4.1 Introduction

Dans les **Équipements CPE** dotés d'une pile IP, l'**Hôte ECI** fournit un service d'accès à Internet pour le compte des **Clients ECI**. Ces derniers peuvent envoyer des messages avec le protocole UDP/IP et ouvrir des connexions TCP/IP avec des homologues en mode **Client ECI** et serveur à l'aide des **Hôtes ECI.** Les noms d'**Hôte ECI** peuvent être associés à des adresses IP utilisant les services DNS disponibles des **Hôtes ECI.**

La sécurité des services fournis se limite à la sécurité logicielle générique de l'**Équipement CPE** lui-même. En d'autres termes, si le logiciel de l'**Équipement CPE** extérieur à l'**Hôte ECI** est compromis, n'importe quel trafic IP peut être altéré.

L'API des **Clients ECI** relative à la connectivité IP est fondée sur le paradigme du socket BSD utilisé par de nombreux systèmes d'exploitation actuels.

La définition de cette API comporte quatre parties:

1) Sockets IP **ECI** de base et fonctionnalité DNS (§ 9.4.4.3).

2) Communication UDP/IP avec un socket IP **ECI** (§ 9.4.4.4).

3) Communication TCP/IP avec un socket IP **ECI** (§ 9.4.4.5).

4) Communication HTTP(S) avec des services HTTP de l'**Hôte ECI** (§ 9.4.4.6).

9.4.4.2 Spécifications de base

Les **Hôtes ECI** dotés d'une capacité de connexion IP mettront en œuvre le protocole IP [IETF RFC 791], y compris l'IPv6 ([IETF RFC 8200] et ses mises à jour applicables. Ce protocole permettra de résoudre le nom de l'**Hôte ECI** à des adresses IP à l'aide du système DNS conformément à [IETF RFC 1034], [IETF RFC 1035] et leurs mises à jour.

Afin de fournir un protocole de messages non fiable, bref et simple, l'**Hôte ECI** prendra en charge UDP sur IP conformément à [IETF RFC 768] et à ses mises à jour applicables. Afin d'assurer un échange de messages fiable faisant appel à une connexion avec contrôle des flux, l'**Hôte ECI** prendra en charge TCP sur IP conformément à [IETF RFC 793] et à ses mises à jour applicables.

L'**Hôte ECI** n'est pas obligé de prendre en charge la multidiffusion UDP en mode transmission ou réception.

9.4.4.3 Sockets IP ECI

9.4.4.3.1 Généralités

Les **Clients ECI** peuvent ouvrir un socket IP **ECI** pour envoyer et recevoir des communications avec les protocoles TCP et IP.

NOTE – Le terme "socket" évoque les sockets BSD utilisés à l'origine dans de nombreux systèmes d'exploitation. Les sockets IP ECI sont conceptuellement similaires mais possèdent des propriétés spécifiques différentes de celles des sockets BDS. Plus précisément, leur comportement est entièrement asynchrone.

Les sockets IP **ECI** sont les points d'extrémité de la communication IP. Les **Clients ECI** peuvent ouvrir un socket en identifiant le numéro de port local et en acceptant les **Requêtes** de connexion entrantes (fonctionnent comme un serveur TCP/IP). Il est possible de fermer les sockets. Dans ce cas, les connexions associées ou le serveur sont fermés. L'adresse IP du nom d'hôte d'un homologue peut être résolue à l'aide des services DNS de l'**Hôte ECI**.

Les messages disponibles sont répertoriés dans le Tableau 9.4.4.3.1-1.

Tableau 9.4.4.3.1-1 – Messages des sockets IP

Message	Type	Sens	Étiquette	Description
reqIpSocket	A	C→H	0x0	Ouvre un socket IP **ECI**.
reqIpClose	A	C→H	0x1	Ferme un socket IP **ECI**.
reqIpAddrinfo	A	C→H	0x2	Obtient l'adresse d'un **Hôte ECI** (distant).

Les types de structure de ces API sont définis dans le § 9.3.

Définitions des types pour l'API des sockets IP:

```
typedef struct Addrinfo {
   ushort addressType;       /* adresse IPv4 ou IPv6*/
   uchar ipAddress[16];      /* adresse IP elle-même */
   ushort port;              /* numéro de port - si pertinent
*/
} Addrinfo;
```

Définitions des champs:

addressTyp: ushort.	Voir le Tableau 9.4.4.3.4-1. Seules les valeurs ProtPrefIPv4 ou ProtPrefIPv6 sont autorisées. Ce champ définit la longueur de l'adresse de l'hôte hostAddress à 4 ou 16 octets (voir la NOTE).
ipAddress: uchar[16]	4 ou 16 octets correspondant à la représentation en octets (dans l'ordre du réseau) d'une adresse IPv4 ou IPv6 respectivement. Les adresses IPv4 utiliseront les 4 premiers octets de ce paramètre.
port: ushort	Numéro de port du socket auquel se connecter (ce champ peut être inutilisé).
NOTE – **ProtPrefIPv4** ou **ProtPrefIPv6** sont définis dans le Tableau 9.4.4.3.4-1.	

9.4.4.3.2 Message reqIpSocket

C→H reqIpSocket(uchar **source,** ushort **sourcePort,** ushort **protocol)** →
H→C resIpSocket(uchar **socketId)**

- Ce message ouvre un socket pour une communication TCP ou UDP sur une adresse IP et un port locaux.

Définitions des paramètres de la requête:

source: uchar	Voir le Tableau 9.4.4.3.2-1: spécifie l'adresse IP de l'hôte ECI à utiliser pour le socket local (à privilégier si plusieurs adresses IP sont attribuées). Si l'adresse IP concernée n'est pas identifiable, l'hôte ECI choisira une alternative convenable.
sourcePort: ushort.	Adresse du port du point d'extrémité local de la connexion IP. Une valeur égale à 0x0000 signifiera que l'hôte ECI allouera une adresse de port libre au socket. Les valeurs inférieures à 1024 ne sont pas autorisées.
Protocol: ushort	Voir le Tableau 9.4.4.3.2-2: spécifie le protocole utilisé pour le socket. Le choix d'IPv4 ou IPv6 sera spécifique.

Tableau 9.4.4.3.2-1 – Paramètres de la source IP

Nom	Valeur	Description
IpSourceAny	0x00	Adresse IP par défaut de l'**Hôte ECI**.
IpSourceWan	0x01	Adresse IP de l'**Hôte ECI** utilisée pour une communication WAN (Internet).
IpSourcePriv	0x02	Adresse IP de l'**Hôte ECI** utilisée pour le trafic IP privé sur un canal de protocole IP propriétaire.
IpSourceLan	0x03	Adresse IP de l'**Hôte ECI** utilisée pour la communication sur réseau local.
RFU	Autre	Réservé à une utilisation future.

Tableau 9.4.4.3.2-2 – Paramètres du protocole IP

Nom	Valeur	Description
SockProtUdpIPv4	0x0001	UDP/IP avec IPv4.
SockProtUdpIPv6	0x0002	UDP/IP avec IPv6.
SockProtUdpIPany	0x0003	UDP/IP avec IPv4 ou v6.
SockProtTcpClientIpv4	0x0005	TCP/IP avec IPv4, mode client (uniquement pour initialiser les connexions).
SockProtTcpClientIpv6	0x0006	TCP/IP avec IPv6, mode client (uniquement pour initialiser les connexions).
SockProtTcpClientIpany	0x0007	TCP/IP avec IPv4 ou v6, mode client (uniquement pour initialiser les connexions).
SockProtTcpServerIpv4	0x0009	TCP/IP avec IPv4, mode serveur (pour accepter les connexions entrantes).
SockProtTcpServerIpv6	0x000A	TCP/IP avec IPv6, mode serveur (pour accepter les connexions entrantes).
SockProtTcpServerIpany	0x000B	TCP/IP avec IPv4 ou v6, mode serveur (pour accepter les connexions entrantes).
RFU	autre	Réservé à une utilisation future.

Définitions des paramètres de la réponse:

SocketId: uchar.	Identificateur du socket ouvert.

Description sémantique:

- Immédiatement après l'initialisation, la **Réponse** peut être bloquée jusqu'à l'initialisation réussie de l'adresse IP de l'**Hôte ECI**. Des chiffres de performance sont proposés dans le document [b-UIT-T J Suppl. 7].

Préconditions de la requête:

1) Le nombre maximal de sockets que le **Client ECI** est autorisé à demander ne sera pas dépassé.

2) Les paramètres de source, de port source et de protocole sont valables.

Postconditions de la réponse:

1) Le socket est ouvert ou la **Réponse** renvoie une erreur.

Les codes d'erreur relatifs à l'ouverture des sockets sont répertoriés dans le Tableau 9.4.4.3.2-3.

Tableau 9.4.4.3.2-3 – Codes d'erreur de la requête resIpSocket

Nom	Description
ErrIpSourceProt	Voir le Tableau 9.4.4.7-1.
ErrIpNoSockets	
ErrIpProtNotAvail	
ErrIpPortNotAvail	

9.4.4.3.3 Message reqIpClose

C→H reqIpClose(uchar **socketId)** →
H→C resIpClose(uchar **socketId)**

- Ferme le socket IP et la connexion associée. Toutes les communications en instance à destination ou en provenance du socket peuvent être perdues.

Définitions des paramètres de la requête:

socketId: uchar	Identificateur du socket à fermer.

Définitions des paramètres de la réponse:

socketId: uchar.	Identificateur du socket fermé.

Description sémantique:

- Cette **Requête** ferme le socket et la connexion IP associée. Il revient à l'**Hôte ECI** d'envoyer les messages de déconnexion appropriés à l'homologue impliqué dans la communication, le cas échéant. L'envoi de la **Réponse** ne requiert pas l'exécution réussie de leur envoi. Les sockets auxquels aucune connexion n'est associée seront fermés également.

Préconditions:

1) Le socket existe et est ouvert.

Postconditions:

1) Le socket est fermé et ne peut plus servir à la communication (sauf réaffectation par le message reqIpSocket).

Les codes d'erreur relatifs à la fermeture des sockets sont répertoriés dans le Tableau 9.4.4.3.3-1.

Tableau 9.4.4.3.3-1 – Codes d'erreur de la requête resIpClose

Nom	Description
ErrIpSocketNotOpen	Voir le Tableau 9.4.4.7-1.

9.4.4.3.4 Message reqIpAddrInfo

C→H reqIpAddrinfo(uint **hostnameLenth,** char **hostname[],** uchar **protPref)** →
H→C resIpAddrinfo(Addrinfo **ipaddress)**

- Ce message fournit les informations relatives à l'adresse IP requises pour définir l'adresse de l'**Hôte ECI** à l'aide du protocole préféré (protPref) et renvoie l'adresse de l'**Hôte ECI**. Si nécessaire, le protocole utilisera les services DNS de l'**Hôte ECI** pour résoudre la **Requête**.

Définitions des paramètres de la requête:

hostNameLength: uint	Longueur du champ de nom (en octets).
hostname: char[]	Nom de l'hôte IP à résoudre. Peut prendre la forme d'une notation DOD IPv4 [IETF RFC 952], d'une notation employant le signe deux points IPv6 [IETF RFC 8200] ou du nom de l'hôte lui-même [IETF RFC 1123].
protPref: uchar	Indique la préférence relative au protocole IP telle que définie dans le Tableau 9.4.4.3.4-1.

Tableau 9.4.4.3.4-1 – Paramètres de préférence relative au protocole IP

Nom	Valeur	Description
ProtPrefIpv4	0x1	Une adresse IPv4 sera renvoyée.
ProtPrefIPv6	0x2	Une adresse IPv6 sera renvoyée.
ProtPrefAny	0x3	Une adresse IPv4 ou IPv6 sera renvoyée.
RFU	autre	Réservé à une utilisation future.

Définitions des paramètres de la réponse:

Ipaddress: Addrinfo	Adresse IP de l'**Hôte ECI**. Le champ de port n'est pas défini.

Description sémantique:

- Cette **Requête** utilise les services DNS de l'**Hôte ECI** pour traduire le nom d'hôte fourni en représentation binaire d'adresse d'hôte. L'absence temporaire d'accès aux services DNS (par exemple, au démarrage de l'**Équipement CPE**) pourra entraîner des ajournements. L'**Hôte ECI** veillera à ce qu'un délai d'attente approprié soit observé (afin que le **Client ECI** reçoive toujours la **Réponse**).

Postconditions de la réponse:

1) Adresse d'hôte résolue ou erreur.

Les codes d'erreur relatifs à la fermeture des sockets sont répertoriés dans le Tableau 9.4.4.3.4-2.

Tableau 9.4.4.3.4-2 – Codes d'erreur de la requête resIpAddrInfo

Nom	Description
ErrIpHostUnknown	Voir le Tableau 9.4.4.7-1.
ErrIpHost	
ErrDnsOffline	

9.4.4.4 UDP/IP ECI

9.4.4.4.1 Généralités

Les **Clients ECI** enverront et recevront des datagrammes UDP via un socket UDP/IP ouvert. Les messages concernés sont définis dans le Tableau 9.4.4.4.1-1.

Tableau 9.4.4.4.1-1 – Messages des sockets UDP/IP

Message	Type	Sens	Étiquette	Description
reqIpUdpSendMsg	A	C→H	0x3	Envoie un message au port UDP de l'homologue.
reqIpUdpRecvMsg	A	C→H	0x4	Reçoit un message du port UDP de l'homologue.

9.4.4.4.2 Message reqIpUdpSendMsg

C→H reqIpUdpSendMsg(uchar **socketId,** Addrinfo **peer,** uint **datagramLength,** byte **datagram[]) →**
H→C resIpUpdSendMsg(uchar **socketId)**

Ce message envoie un datagramme UDP à un homologue (adresse IP, port IP).

Définitions des paramètres de la requête:

socketId: uchar	Longueur du champ de nom (en octets).
peer: Addrinfo	Homologue (adresse IP, numéro de port IP) destinataire du datagramme.
datagramLength: uint	Longueur (en octets) du datagramme.
datagram: byte[]	Contenu du datagramme (octets dans l'ordre du réseau).

Définitions des paramètres de la réponse:

socketId: uchar	Socket d'émission de la **Requête** correspondante.

Description sémantique:

- Le datagramme est envoyé à l'aide du protocole UDP; l'adresse et le port de l'hôte IP du socket sont envoyés à l'homologue.

Préconditions de la requête:

1) Le socket a été ouvert pour le protocole UDP à l'aide de la même structure d'adresse que celle de l'homologue.

Postconditions:

1) Le datagramme est envoyé (mais peut être perdu).

Les codes d'erreur relatifs à l'envoi de datagrammes UDP sont répertoriés dans le Tableau 9.4.4.4.2-1.

Tableau 9.4.4.4.2-1 – Codes d'erreur de la requête resIpUdpSendMsg

Nom	Description
ErrIpUdpProtMismatch	Voir le Tableau 9.4.4.7-1.
ErrIpUdpSocketNot	
ErrIpUdpTooLong	
ErrIpUdpIpOffline	

9.4.4.4.3 Message reqIpUdpRecvMsg

C→H reqIpUdpRecvMsg(uchar **socketId) →**
H→C resIpUdpRecvMsg(uchar **socketId,** Addrinfo **peer,** uint **datagramLength,** byte **datagram[])**

- Ce message permet au **Client ECI** d'envoyer à l'**Hôte ECI** une requête de réception d'un datagramme UDP en provenance d'un homologue (c'est-à-dire nom d'hôte, port) envoyé au socket avec **SocketId**.

Définitions des paramètres de la requête:

socketId: uchar	Socket (impliquant un numéro de port et une adresse d'hôte) sur lequel la réception d'un datagramme UDP est attendue.

Définitions des paramètres de la réponse:

socketId: uchar	Longueur du champ de nom (en octets).
peer: Addrinfo	Adresse IP + numéro de port de la source du datagramme (homologue)
datagramLength: uint	Longueur (en octets) du datagramme.
datagram: byte[]	Contenu du datagramme (octets dans l'ordre du réseau).

Description sémantique:

- Il est possible de recevoir un datagramme sur le socket, auquel cas une **Réponse** est renvoyée.

NOTE 1 – La fermeture du socket mettra fin aux **Requêtes reqIpUdpRecvMsg** en instance.

NOTE 2 – L'émission de plusieurs requêtes **reqIpUdpRecvMsg** avant la réception des **Réponses** correspondantes sur le même socket est autorisée mais l'**Hôte ECI** n'est pas obligé de prendre en charge la mise en file d'attente de plus de cinq **Requêtes** de ce type.

Préconditions de la requête:

- Un socket a été ouvert pour le protocole UDP.

Postconditions de la réponse:

- Le datagramme est envoyé (mais peut être perdu).

Les codes d'erreur relatifs à la réception de datagrammes UDP sont répertoriés dans le Tableau 9.4.4.4.3-1.

Tableau 9.4.4.4.3-1 – Codes d'erreur de la requête resIpUdpRecvMsg

Nom	Description
ErrIpUdpSocketNot	Voir le Tableau 9.4.4.7-1.

9.4.4.5 TCP/IP ECI

9.4.4.5.1 Généralités

Les **Clients ECI** peuvent envoyer et recevoir des messages sur une connexion TCP/IP ouverte lors de la création d'un socket mettant en place une séquence de flux d'octets bidirectionnel sans erreur entre le **Client ECI** local et un service d'homologue distant ou inversement. Cette possibilité permet aux **Clients ECI** de jouer le rôle de serveur pour les **Requêtes** de canaux émanant d'autres parties (en général des applications LAN). Les messages sont répertoriés dans le Tableau 9.4.4.5.1-1.

Tableau 9.4.4.5.1-1 – Messages relatifs aux sockets TCP/IP

Message	Type	Sens	Étiquette	Description
reqIpTcpConnect	A	C→H	0x5	Le client TCP se connecte au serveur TCP homologue.
reqIpTcpSend	A	C→H	0x6	Envoie des données à l'homologue connecté.
reqIpTcpRecv	A	C→H	0x7	Reçoit des données de l'homologue connecté.
reqIpTcpAccept	A	C→H	0x8	Le serveur TCP homologue accepte la connexion en provenance du client TCP homologue.

9.4.4.5.2 Message reqIpTcpConnect

C→H reqIpTcpConnect(uchar **socketId,** Addrinfo **peer)** →
H→C resIpTcpConnect(uchar **socketId)**

- Ce message demande à l'**Hôte ECI** d'ouvrir une connexion à partir d'un socket TCP ouvert vers l'homologue en utilisant le protocole du socket.

Définitions des paramètres de la requête:

socketId: uchar	Socket (impliquant un numéro de port et une adresse d'hôte) à partir duquel une connexion TCP sera établie.
peer: Addrinfo	Adresse IP et port IP de l'homologue avec lequel la connexion doit être ouverte.

Définitions des paramètres de la réponse:

socketId: uchar	Identificateur du socket de la **Requête**.

Description sémantique:

- L'hôte local s'efforcera d'ouvrir une connexion TCP avec l'homologue (adresse IP, port IP) à partir du socket local.

Préconditions:

- Le socket a été ouvert pour le protocole TCP avec le même type d'adresse IP (IPv4 ou IPv6) que peerAddressType.

Postconditions:

- Une connexion TCP est établie ou une condition d'erreur est renvoyée.

Les codes d'erreur relatifs aux connexions via TCP et IP sont répertoriés dans le Tableau 9.4.4.5.2-1.

Tableau 9.4.4.5.2-1 – Codes d'erreur de la requête resIpTcpConnect

Nom	Description
ErrIpTcpProtMismatch	Voir le Tableau 9.4.4.7-1.
ErrIpTcpSockNot	
ErrIpTcpIpOffline	
ErrIpTcpConnRefused	
ErrIpTcpConnTimeout	

9.4.4.5.3 Message reqIpTcpSend

C→H reqIpTcpSend(uchar **socketId,** bool **more,** uint **dataLen,** byte **data[])** →
H→C resIpTcpSend(uchar **socketId,** uint **actLen)**

- Ce message envoie des données via TCP sur un socket connecté à ce protocole.

Définitions des paramètres de la requête:

socketId: uchar	Socket (impliquant un numéro de port et une adresse d'hôte) utilisé pour envoyer les données à l'homologue.
more: bool	Indique si les données présentes et précédentes doivent être envoyées à l'homologue immédiatement (**more=False**) ou si d'autres données figureront dans les **Requêtes** reqipTcpSend suivantes (**more=True**).
dataLen: uint	Quantité de données à envoyer.
data[]**:** byte	Données à envoyer.

Définitions des paramètres de la réponse:

socketId: uchar	Identificateur du socket émetteur de l'envoi.
actLen: uint	Nombre réel d'octets envoyés avec succès.

Description sémantique:

- L'hôte local enverra les **données** à l'homologue connecté sur un socket TCP/IP avec **socketID**.

Préconditions de la requête:

1) Le socket est en mode TCP/IP connecté.

Postconditions de la réponse:

1) Si actLen n'est pas égal à dataLen, une condition d'erreur s'appliquera.

Les codes d'erreur relatifs à l'envoi de paquets TCP sont répertoriés dans le Tableau 9.4.4.5.3-1.

Tableau 9.4.4.5.3-1 – Codes d'erreur de la requête resIpTcpSend

Nom	Description
ErrIpTcpSockNot	Voir le Tableau 9.4.4.7-1.
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.4 Message reqIpTcpRecv

C→H reqIpTcpRecv(uchar **socketId,** uint **maxDataLen)** →
H→C resIpTcpRecv(uchar **socketId,** uint **dataLength,** byte **data[])**

- Ce message reçoit des données via TCP sur un socket connecté à ce protocole.

Définitions des paramètres de la requête:

socketId: uchar	Socket (impliquant un numéro de port et une adresse d'hôte) utilisé pour recevoir les données en provenance de l'homologue.
maxDataLen: uint	Quantité maximale de données à recevoir.

Définitions des paramètres de la réponse:

socketId: uchar	Identificateur du socket émetteur du message de réception.
dataLength: uint	Nombre d'octets de données reçus de l'homologue.
data[]**:** byte	Données reçues de l'homologue.

Description sémantique:

- L'hôte local reçoit des **données** en provenance de l'homologue sur un socket connecté à TCP/IP avec **socketID**.

Préconditions de la requête:

1) Le socket est un socket TCP.

Postconditions de la réponse:

1) Toutes les données disponibles jusqu'à la longueur définie sont renvoyées jusqu'à la valeur du champ **maxDataLen** de la **Requête**. En l'absence de données, la **Réponse** sera bloquée jusqu'à ce que la connexion soit fermée, que la connexion TCP soit considérée temporairement indisponible ou que la connexion locale au réseau IP soit perdue.

Les codes d'erreur relatifs à la réception de paquets TCP sont répertoriés dans le Tableau 9.4.4.5.4-1.

Tableau 9.4.4.5.4-1 – Codes d'erreur de la requête resIpTcpRecv

Nom	Description
ErrIpTcpSockNot	Voir le Tableau 9.4.4.7-1.
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.5 Message reqIpTcpAccept

C→H reqIpTcpAccept(uchar **socketId)** →
H→C resIpTcpAccept(uchar **socketId,** uchar **newSocketId,** Addrinfo **peer)**

- Ce message accepte une **Requête** de connexion entrante sur un socket de serveur TCP. En attendant la connexion, les **Requêtes** seront prises en charge jusqu'au nombre maximal défini par la mise en œuvre de l'**Hôte ECI**. Des exigences de performance du serveur TCP sont proposées dans le document [b-UIT-T J Suppl. 7].

Définitions des paramètres de la requête:

socketId: uchar	Socket (impliquant un numéro de port et une adresse d'hôte) utilisé pour recevoir des **Requêtes** de connexion.

Définitions des champs de message:

socketId: uchar	Identificateur du socket émetteur de la requête.
newSocketId: uchar	Identificateur du socket de la connexion nouvellement ouverte avec l'homologue émetteur de la **Requête** de connexion. L'adresse et le port de l'hôte sont hérités du socket associé à **socketId**.
peer: Addrinfo	Adresse IP + port IP de l'homologue lors de la connexion.

Description sémantique:

- L'**Hôte ECI** local attend les **Requêtes** de connexion TCP entrantes sur l'adresse/port IP spécifiés lors de la création du socket et ouvre un nouveau socket connecté prenant en charge la **Requête** de connexion entrante (ou en instance). En l'absence de **Requête** entrante ou si le socket du serveur est fermé, aucune **Réponse** ne suit.

Préconditions de la requête:

1) Le socket est un socket de serveur TCP.

Postconditions de la réponse:

1) Un nouveau socket doté d'une connexion TCP/IP ouverte est renvoyé suite à une **Requête** de connexion disponible adressée au socket du serveur ou une erreur survient.

Les codes d'erreur relatifs à l'acceptation des connexions TCP sont répertoriés dans le Tableau 9.4.4.5.5-1.

Tableau 9.4.4.5.5-1 – Codes d'erreur de la requête resIpTcpAccept

Nom	Description
ErrIpTcpListSockNot	Voir le Tableau 9.4.4.7-1.
ErrIpTcpNoMoreSockets	

9.4.4.6 API des services HTTP(S) GET

9.4.4.6.1 Généralités

L'**Hôte ECI** fournira des requêtes HTTP(S) Get de base dans le but de récupérer des ressources sur un serveur HTTP basé sur IP pour le compte du client. Cette possibilité permet au **Client ECI** de récupérer des ressources web (fichiers) sur des serveurs Internet. Le protocole HTTPS peut servir, entre autres, à récupérer des ressources basées sur des API web telles que des données d'importation et d'exportation (voir les § 9.7.2 et 7.8.4.2).

La sécurité est assurée par le protocole HTTPS (TLS) de la mise en œuvre du protocole TLS de l'**Équipement CPE** sous-jacent.

NOTE – En général, cette sécurité ne doit pas servir à protéger l'intégrité des contenus des **Clients ECI** mais peut être utilisée pour empêcher les dénis de services distribués (DDOS) et d'autres tentatives opportunistes de manipulation des **Clients ECI.**

L'**Hôte ECI** mettra à la disposition des **Clients ECI** la quantité minimum de ressources permettant d'émettre des requêtes HTTP Get. Des valeurs sont proposées dans le document [b-UIT-T J Suppl. 7].

Les messages relatifs à l'API HTTP(S) Get sont répertoriés dans le Tableau 9.4.4.6.1-1.

Tableau 9.4.4.6.1-1 – Messages relatifs à l'API HTTP GET

Message	Type	Sens	Étiquette	Description
reqHttpGetFile	A	C→H	0x0	Exécute une requête HTTP GET sur une adresse URL et stocke le résultat dans un fichier.
reqHttpGetData	A	C→H	0x1	Exécute une requête HTTP GET sur une adresse URL et transmet le résultat au Client sous forme de données.

9.4.4.6.2 Spécifications applicables

NOTE – Les spécifications ci-dessous ne constituent pas une partie essentielle de la sécurité ECI, comme indiqué dans le § 9.4.4.6.1.

La mise en œuvre des protocoles HTTP et HTTPS dans le cadre de la mise en œuvre de l'API du **Client ECI** sera conforme à HTTP1.1 [IETF RFC 7230] et [IETF RFC 7231].

La mise en œuvre du protocole de sécurité de la couche de transport (TLS) utilisé pour fournir des services HTTP au **Client ECI** sera conforme à TLS 1.3 [IETF RFC 8446]. Pour la rétrocompatibilité, TLS 1.2 doit être pris en charge conformément aux contraintes de TLS 1.3 et aux règles suivantes:

1) TLS 1.2, voir [IETF RFC 5246].
2) TLS AES-GCM, voir [IETF RFC 5288].
3) Extensions du protocole TLS, voir [IETF RFC 6066].
4) PKIX/X.509 [IETF RFC 5280] + mises à jour [IETF RFC 6818].

Toutes les mises en œuvre de TLS 1.2 prendront en charge les systèmes cryptographiques définis dans [IETF RFC 5246]:

1) TLS_RSA_WITH_AES_128_CBC_SHA256.
2) TLS_DHE_RSA_WITH_AES_128_GCM_SHA256.

D'autres systèmes cryptographiques pour TLS 1.2 pourront être pris en charge conformément aux contraintes de TLS 1.3.

Le choix des systèmes cryptographiques de TLS 1.2 suit les règles ci-après:

1) TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 doit être le système cryptographique par défaut.
2) La priorité sera donnée aux systèmes cryptographiques AEAD.
3) La priorité sera donnée à l'échange de clé fondé sur DHE.
4) Les clés d'une longueur supérieure à 128 bits ne seront pas prioritaires.
5) La norme 3DES ne doit pas être utilisée.
6) L'algorithme RC4 ne sera pas utilisé (comme spécifié dans [W3C PNG]).
7) L'algorithme MD5 ne sera pas utilisé (comme spécifié dans [IETF RFC 6151]).

Les règles de traitement suivantes s'appliquent:

1) La version minimum requise par toutes les entités **ECI** sera TLS 1.2.
2) Les versions SSL 2.0 et 3.0 ne seront pas utilisées.
3) La renégociation ne sera pas utilisée.
4) La compression ne doit pas être utilisée (acceptable avec le mode GCM).
5) Les nombres premiers de DH//DHE compteront au moins 1 024 bits et seront vérifiés lors de la prise de contact avec le protocole TLS.
6) La vérification des **Certificats** et des hôtes respectera les exigences en matière de PKIX [IETF RFC 5280] et [IETF RFC 6125].

Les certificats racine utilisés pour authentifier l'homologue de la connexion TLS doivent être fondés sur une liste actualisée, par exemple https://cabforum.org/browser-os-info/.

Les **Équipements CPE** doivent prendre en charge un moyen permettant au **Fabricant d'équipement CPE** de supprimer ou de rejeter des certificats racine après fabrication. Pour ce faire, on peut recourir à un mécanisme de mise à niveau de micrologiciel ou, de préférence, à un mécanisme spécifique de mise à jour du certificat racine qui pourrait permettre des mises à jour plus rapides. Un **Fabricant d'équipement CPE** peut choisir de supprimer ou de rejeter un certificat racine obligatoire dans l'**Équipement CPE** à la suite d'une menace de sécurité. Les **Équipements CPE** devraient prendre en charge un moyen d'ajouter en toute sécurité de nouveaux certificats racine après fabrication afin de maintenir l'interopérabilité avec les serveurs dans le temps.

Les règles de traitement décrites par le Forum CA/Browser [b-CA Browser] et [b-NIST SP 800-52r2] fournissent des indications supplémentaires.

NOTE – Dans un souci d'interopérabilité, les serveurs HTTP désignés pour fournir des services HTTP aux **Clients ECI** devront prendre en charge des modes et des options compatibles, ainsi que les recommandations applicables définies ici pour le client HTTP.

9.4.4.6.3 Messages reqHttpGetFile et reqHttpGetData

C→H **reqHttpGetFile(**filename **fname** ;char **url[]**, char **userAgent[]**; uint **redirs**, uint **timeout**) → H→C **resHttpGetFile**(uint **httpStatus**)

C→H **reqHttpGetData(**char **url[]**, **userAgent[]**; uint **redirs**, uint **timeout**) → H→C **resHttpGetData**(uint **httpStatus,** byte **data[]**)

- Ce **message** demande à l'**Hôte ECI** d'effectuer une requête HTTP de récupération d'un fichier et de renvoyer le statut HTTP une fois la tâche exécutée.
- La requête resHttpGetFile renvoie la ressource sous forme de fichier dans le système de fichiers du Client.
- La requête resHttpGetData renvoie la ressource sous forme de données de message d'une taille limitée.

Définition des paramètres de la requête:

fname: fileName	Nom du fichier où le résultat (post-données) de la requête est stocké par l'**Hôte ECI**. Les données existantes seront effacées.
url: char[]	Adresse URL avec codage UTF 8 [IETF RFC 7230]. Il est possible de spécifier des numéros de ports non standard dans l'adresse URL. Le protocole TLS sera utilisé pour les adresses URL conformes à la "structure d'identificateur URI https" traitée dans [IETF RFC 7230].
userAgent: char[]	Spécifie le champ d'entête User-Agent à utiliser comme en-tête HTTP. Les **Clients ECI** peuvent indiquer une valeur spécifique anticipée par le serveur HTTP de l'adresse **url** (voir la NOTE).
redirs: unit	Nombre maximum de redirections autorisé pour traiter la requête. Des chiffres de performance minimum des redirections (**redirs**) sont proposés dans le document [b-UIT-T J Suppl. 7].
timeout: unit	Délai d'exécution de la requête HTTP en millisecondes. En cas de dépassement, la requête sera abandonnée et une erreur de temporisation sera renvoyée dans la **Réponse**.
NOTE – Il est déconseillé d'utiliser le champ User-Agent comme mécanisme de contrôle d'accès ou de sélection de la ressource; l'usage prévu est défini dans [IETF RFC 7231].	

Définition des paramètres de la réponse:

httpStatus: uint	Valeur du statut HTTP.
data[]: byte	Données du résultat de la requête HTTP Get dans l'ordre du réseau. La taille maximale est limitée par la taille du tampon du message.

Sémantique détaillée:

- L'**Hôte ECI** veillera à ce que les requêtes HTTP prennent en charge des fichiers courants et des types de médias très divers. Il est déconseillé d'inclure le champ d'en-tête Accept dans

l'en-tête de requête HTTP. S'il y figure, les types de codage de contenus MIME suivants seront acceptables pour récupérer la ressource: application/octet-stream, application/json, image/jpeg, image/png, image/gif, text/plain, text/html, text/css, text/xml et text/javascript.

- L'**Hôte ECI** veillera à ce que l'en-tête de requête HTTP Accept-Encoding indique que les codages de contenus suivants sont acceptables: gzip.

Postconditions de la réponse:

1) La ressource se trouvant à l'adresse **url** a été récupérée et stockée dans un nom de fichier **fname** (pour **resHttpGetFile**) ou renvoyée sous forme de données (pour **rerHttpGetData**) ou bien une erreur s'est produite.

Les codes d'erreur relatifs aux requêtes resHttpGetFile et resHttpGetData sont répertoriés dans le Tableau 9.4.4.6.3-1.

Tableau 9.4.4.6.3-1 – Codes d'erreur des requêtes resHttpGetFile et resHttpGetData

Nom	Description
ErrHttpGetNoSockets	Voir le Tableau 9.4.4.6.4-1.
ErrHttpGetProtNotAvail	
ErrHttpGetPortNotAvail	
ErrHttpHostUnknown	
ErrHttpDnsOffline	
ErrHttpIpOffline	
ErrHttpTimeout	
ErrHttpGetFSFailure	
ErrHttpGetFSExceeded	
ErrHttpGetTlsAuth	
ErrHttpGetRedir	
ErrHttpGetData	

9.4.4.6.4 Codes d'erreur de l'API HTTP GET

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.4.6.4-1.

Tableau 9.4.4.6.4-1 – Codes d'erreur des API HTTP GET

Nom	Valeur	Description
ErrHttpGetNoSockets	-257	Voir la valeur correspondante des codes d'erreur dans le Tableau 9.4.4.7-1 pour l'API des sockets IP.
ErrHttpGetProtNotAvail	-258	
ErrHttpGetPortNotAvail	-259	
ErrHttpHostUnknown	-261	
ErrHttpDnsOffline	-263	
ErrHttpIpOffline	-267	
ErrHttpTimeout	-270	La requête HTTP n'a pas pu se terminer dans le délai fixé dans la requête.
ErrHttpGetFSFailure	-512	La valeur +256 correspond à la valeur des codes d'erreur figurant dans le Tableau 9.4.5.5-1 pour l'API du système de fichiers.
ErrHttpGetFSExceeded	-514	
ErrHttpGetTlsAuth	-768	Le protocole TLS n'a pas réussi à authentifier le serveur ou les données.
ErrHttpGetRedir	-784	Nombre de redirections dépassé.
ErrHttpError	-785	La ressource n'a pas pu être récupérée sur le serveur. Le code d'erreur HTTP indique le motif.
ErrHttpGetData	-786	Les données de la ressource dépassaient la valeur maximale du champ de longueur des données.

9.4.4.7 Codes d'erreur de l'API des sockets IP

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.4.7-1.

Tableau 9.4.4.7-1 – Codes d'erreur des API des sockets IP

Nom	Valeur	Description
ErrIpSourceProt	-256	Combinaison de source et de protocole non valide.
ErrIpNoSockets	-257	Plus de sockets disponibles.
ErrIpProtNotAvail	-258	Protocole indisponible.
ErrIpPortNotAvail	-259	Port demandé indisponible.
ErrIpSocketNotOpen	-260	Socket non ouvert.
ErrIpHostUnknown	-261	**Hôte ECI** inconnu.
ErrIpHost	-262	**Hôte ECI** connu mais aucune adresse disponible (pour le type d'adresse IP spécifié).
ErrDnsOffline	-263	Service DNS hors ligne, peut-être temporairement.
ErrIpUdpProtMismatch	-264	L'adresse de l'homologue ne correspond pas au protocole du socket.
ErrIpUdpSockNot	-265	Le socket n'est pas un socket UDP.
ErrIpUdpTooLong	-266	Datagramme trop long pour un seul message UDP.
ErrIpUdpIpOffline	-267	Connexion IP hors ligne (impossible d'atteindre l'homologue).
ErrIpTcpProtMismatch	-268	L'adresse de l'homologue ne correspond pas au protocole du socket.
ErrIpTcpSockNot	-269	Le socket n'est pas un socket TCP.
ErrIpTcpIpOffline	-258	Pas de connexion Internet locale pour le moment.
ErrIpTcpConnRefused	-259	Connexion refusée par l'hôte homologue sur ce port.
ErrIpTcpConnTimeout	-260	Impossible d'obtenir une **Réponse** de l'**Hôte ECI** homologue.
ErrIpTcpClosed	-261	Connexion TCP pas ou plus disponible.
ErrIpTcpListSockNot	-262	Le socket n'est pas un socket de serveur TCP.
ErrIpTcpNoMoreSockets	-263	Une **Requête** de connexion entrante a été reçue, mais l'hôte n'a plus de sockets disponibles.
RFU	Autre	Réservé à une utilisation future.

9.4.5 API d'accès au système de fichiers

9.4.5.1 Introduction

Le **Client ECI** a accès à un système de fichiers privé pour stocker le volume limité des données qui survivront aux cycles de vie du **Client ECI**, aux cycles d'alimentation de l'**Équipement CPE**, aux pannes du système, etc., dans des conditions de fonctionnement normales. La fiabilité doit être au moins égale à celle du système de fichiers normal de l'**Équipement CPE**. En d'autres termes, les défaillances survenant dans des circonstances exceptionnelles risquent de gêner l'**Utilisateur**. Il revient au système de sécurité de gérer le **Client ECI** de manière à ne pas infliger à l'**Utilisateur** la

perte de ses droits d'accès aux contenus. Le système de fichiers n'est pas sûr. En condition normale (c'est-à-dire **Équipement CPE** et **Hôte ECI** non compromis), sa manipulation par d'autres entités que le **Client ECI** désigné et son **Hôte ECI** ne sera pas possible.

La représentation abstraite du système de fichiers est un répertoire uniforme unique. Un service de répertoire de base est disponible. Les fonctions d'accès au système de fichiers sont analogues aux appels du système de fichiers d'Unix/Linux/Posix tels que open, close, write, read, lseek, opendir, readdir et lstat.

Un volume minimum d'espace de stockage du système de fichiers sera à la disposition de chaque **Client ECI** s'il est stocké par l'**Utilisateur**. Un volume est proposé dans le document [b-UIT-T J Suppl. 7].

L'API du système de fichiers comporte trois partitions secondaires:

1) Ouverture et fermeture des fichiers.
2) Lecture et écriture des fichiers, accès aléatoire et suppression de données sélectionnées dans un fichier.
3) Services de répertoire.

Les noms de fichiers seront composés d'une séquence de caractères ASCII 8 bits contenant au moins un et au plus huit des caractères suivants (séparés par une virgule): A-Z, a-z, 0-9, _ et se termineront par un caractère NUL. La définition des noms de fichiers figure dans le Tableau 9.4.5.1-1.

Tableau 9.4.5.1-1 – Structure des noms de fichiers (fileName)

```
typedef char fileName[9];
```

Une fonctionnalité des fichiers journaux permet aux **Clients ECI** d'écrire des volumes limités de données à la manière d'un tampon, c'est-à-dire sans interrompre l'exécution. Le nombre de fichiers journaux par **Client ECI** est défini dans xxx (au minimum deux par client). Ces fichiers peuvent ainsi être journalisés au niveau des applications, retrouvés et analysés post mortem.

9.4.5.2 Ouverture et fermeture des fichiers

9.4.5.2.1 Généralités

La possibilité pour les **Clients ECI** d'ouvrir un fichier en lecture et/ou en écriture génère une poignée de fichier (fileHandle) permettant de fournir des accès en lecture et en écriture ultérieurs. Si un fichier n'existe pas, il est possible de le créer. Le fichier est doté d'une propriété (file location) désignant l'emplacement actuel où il est possible d'y accéder.

Les poignées de fichiers seront gérées par l'**Hôte ECI**. Il ne sera pas possible de réutiliser une poignée de fichier immédiatement après sa fermeture afin d'éviter que le **Client ECI** accède au mauvais fichier du fait d'accès non synchronisés au fichier visé.

Le Tableau 9.4.5.2.1-1 décrit les messages d'ouverture et de fermeture de fichier:

Tableau 9.4.5.2.1-1 – Messages d'ouverture et de fermeture des fichiers

Message	Type	Sens	Étiquette	Description
reqFileOpen	A	C→H	0x0	Ouvre un fichier privé d'un **Client ECI**.
reqFileClose	A	C→H	0x1	Ferme un fichier ouvert.

9.4.5.2.2 Message reqFileOpen

C→H reqFileOpen(fileName **fname,** uint **fileOpenOptions**) →
H→C resFileOpen(uchar **fileHandle)**

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** d'ouvrir un fichier doté de certaines autorisations d'accès.

Définitions des paramètres de la requête:

fname: filename	Nom du fichier à ouvrir.
fileOpenOptions: unit	Mode d'accès pour l'ouverture du fichier. Les valeurs autorisées et leur signification sont définies dans le Tableau 9.4.5.2.2-1.

Tableau 9.4.5.2.2-1 – Options d'ouverture des fichiers

Nom	Bits	Valeur	Description
FileRead	0,1	0b00	Le fichier est ouvert en lecture. La lecture commence au début du fichier.
FileWriteAppend	0,1	0b01	Le fichier est ouvert en écriture. Les écritures suivantes sont ajoutées à la fin du fichier existant. L'emplacement se situe à la fin du fichier.
FileWriteOver	0,1	b11	Le fichier est ouvert en écriture à n'importe quel emplacement. L'emplacement se situe à la fin du fichier.
Non utilisé	0,1	0b10	Non autorisé.
LogFileNo	2	0b0	Fichier normal.
LogFileYes	2	0b1	Fichier journal spécial autorisant les écritures synchrones.
Bits32-2		Autre	Réservé à une utilisation future.

Définitions des paramètres de la réponse:

fileHandle: uchar	Référence (poignée) relative au fichier ouvert.

Postconditions de la requête:

1) Fichier ouvert dans le mode d'accès souhaité ou renvoi d'une erreur. Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.2.2-2.

Tableau 9.4.5.2.2-2 – Codes d'erreur de la requête resfileOpen

Nom	Description
ErrFileNameNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileQuotaExceeded	
ErrFileSystemFailure	

9.4.5.2.3 Message reqFileClose

C→H reqFileClose(uchar **fileHandle)** →
H→C resFileClose()

- Ce message ferme l'accès au fichier ouvert avec **fileHandle**. Les codes d'erreur relatifs à la fermeture des fichiers sont répertoriés dans le Tableau 9.4.5.2.3-1.

Définitions des paramètres de la requête:

fileHandle: uchar	Poignée du fichier à fermer.

Préconditions de la requête:

1) La poignée du fichier est ouverte.

Postconditions de la requête:

1) Les accès ultérieurs à fileHandle échoueront et renverront le code d'erreur ErrFileNotOpen.
2) Les écritures en instance seront conservées (sauf en cas d'erreur).

Tableau 9.4.5.2.3-1 – Codes d'erreur de la requête resfileClose

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3 Accès aux fichiers

9.4.5.3.1 Généralités

Les messages d'accès aux fichiers permettent d'écrire et de lire un fichier ouvert à l'aide d'une poignée de fichier et d'y repositionner l'emplacement de lecture/écriture existant. Les primitives définies correspondent directement aux conventions de Linux/Unix. Les messages définis sont répertoriés dans le Tableau 9.4.5.3.1-1.

NOTE – Les requêtes reqFileWrite et reqFileRead sont très proches des requêtes reqTcpSend et reqTcpRecv.

Tableau 9.4.5.3.1-1 – Messages d'accès aux fichiers

Message	Type	Sens	Étiquette	Description
reqFileWrite	A	C→H	0x2	Ecrit des octets consécutifs à partir de l'emplacement existant dans le fichier.
reqFileRead	A	C→H	0x3	Lit des octets consécutifs à partir de l'emplacement existant dans le fichier.
reqFileSeek	A	C→H	0x4	Repositionne l'emplacement existant dans le fichier.
reqFileRemoveData	A	C→H	0x5	Supprime les données d'un fichier à partir d'un emplacement existant.
callFileDataLog	S	C→H	0x6	Ajoute des données à la fin d'un fichier placé dans la mémoire tampon.

9.4.5.3.2 Message reqFileWrite

C→H reqFileWrite(uchar **fileHandle,** bool **sync,** uint **dataLen,** byte **data[]) →**
H→C resFileWrite(uchar **fileHandle)**

- Ce message écrit des octets dataLen vers le fichier à partir de l'emplacement existant.

Définitions des paramètres de la requête:

fileHandle: uchar	Poignée du fichier cible de l'écriture.
sync: bool	Si la valeur est True, la **Réponse** actualise l'état du système de fichiers avec cette écriture et les précédentes. Si la valeur est False, l'**Hôte ECI** peut placer les **Requêtes** dans la mémoire tampon (elles risquent d'être perdues en cas de défaillance du système).
dataLen: uint	Nombre d'octets à écrire dans le fichier.
data[]: byte	Données à écrire dans le fichier.

Définitions des paramètres de la réponse:

fileHandle: uchar	Poignée du fichier cible de l'écriture.

Préconditions de la requête:

1) Fichier ouvert en écriture (mode FileWriteOver ou FileWriteAppend).
2) Emplacement du fichier où il est possible d'écrire: si le fichier est ouvert en mode FileWriteAppend, l'emplacement se situe à la fin du fichier.
3) Le volume des données à écrire ne provoque pas de problème de quota du système de fichiers.

Postconditions de la requête:

1) L'état du fichier sera actualisé et son emplacement passera de "present" (en instance d'autres opérations de mise en mémoire tampon sur le fichier) à "present+dataLen", sauf en cas d'erreur.

2) Si l'écriture et la synchronisation réussissent, les données sont placées dans la mémoire non volatile du système de fichiers de l'**Hôte ECI**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.3.2-1.

Tableau 9.4.5.3.2-1 – Codes d'erreur de la requête resFileWrite

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.3 Message reqFileRead

C→H reqFileRead(uchar **fileHandle**, uint **dataLen) →**
H→C resFileRead(uchar **fileHandle**, uint **dataRead,** byte **data[])**

- Ce message lit le nombre maximum d'octets dataLen à partir de l'emplacement présent du fichier. Les codes d'erreur relatifs à la lecture des données d'un fichier sont répertoriés dans le Tableau 9.4.5.3.3-1.

Définitions des paramètres de la requête:

fileHandle: uchar	Poignée du fichier objet de la lecture.
dataLen: uint	Nombre maximum d'octets à lire.

Définitions des paramètres de la réponse:

fileHandle: uchar	Poignée du fichier qui a été l'objet de la lecture.
dataRead: uint	Nombre d'octets lus stocké dans **data**.
data[]**:** byte	Données lues.

Préconditions de la requête:

1) Le fichier est ouvert.

Postconditions de la requête:

1) Une erreur s'est produite ou
2) le nombre minimum d'octets **dataLen** ou d'octets demeurant dans le fichier à partir du dernier emplacement est lu et
3) l'emplacement du fichier a été augmenté de la valeur de **dataRead**.
4) Sauf en cas d'erreur, l'emplacement dans le fichier avancera de la valeur de **dataLen** ou sera situé à la fin du fichier.

Tableau 9.4.5.3.3-1 – Codes d'erreur de la requête resFileRead

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3.4 Message reqFileSeek

C→H reqFileSeek(uchar **fileHandle,** int **offset,** uchar **seekPos) →**
H→C resFileSeek(uchar **fileHandle,** int **remOffset)**

- Ce message place un pointeur à une certaine position dans un fichier ouvert et renvoie une partie de son contenu.

Définitions des paramètres de la requête:

fileHandle: uchar	Poignée du fichier dans lequel il faut modifier l'emplacement.
offset: int	Décalage par rapport à l'emplacement de recherche de référence spécifié par **seekPos** qui sera appliqué à l'emplacement dans le fichier.
seekPos: uchar	Voir le Tableau 9.4.5.3.4-1.

Tableau 9.4.5.3.4-1 – Emplacement de référence de la recherche dans le fichier

Nom	Valeur	Description
FileSeekSet	0x00	L'emplacement de référence se situe au début du fichier.
FileSeekCur	0x01	L'emplacement de référence dans le fichier est l'emplacement présent.
FileSeekEnd	0x02	L'emplacement de référence se situe à la fin du fichier.
RFU	Autre	Réservé à une utilisation future.

Définitions des paramètres de la réponse:

fileHandle: uchar	Poignée du fichier pour lequel l'emplacement a été changé.
remOffset: int	Différence entre le décalage spécifié et celui auquel l'emplacement dans le fichier est fixé.

Sémantique détaillée:

- L'emplacement dans le fichier est repositionné et défini dans la description des paramètres de la **Requête**. Il ne se trouve jamais après la fin du fichier ou avant son début. La différence entre le décalage demandé et le décalage réel par rapport à l'emplacement de référence est renvoyée dans le paramètre de résultat **remOffset**. Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.3.4-2.

Préconditions de la requête:

1) Le fichier est ouvert.

Postconditions de la requête:

1) Une erreur s'est produite ou
2) l'emplacement dans le fichier sera fixé comme défini ci-dessus et
3) le paramètre **remOffset** indiquera la différence entre le décalage et l'emplacement réel dans le fichier (comme défini ci-dessus).

Tableau 9.4.5.3.4-2 – Codes d'erreur de la requête resFileRead

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3.5 Message reqFileRemoveData

C→H reqFileRemoveData(uchar **fileHandle,** bool **sync,** uint **dataLen) →**
H→C resFileRemoveData(uchar **fileHandle)**

- Ce message supprime les octets dataLen à partir de l'emplacement présent dans le fichier.

Définitions des paramètres de la requête:

fileHandle: uchar	Poignée du fichier.
sync: bool	Si la valeur est True, la **Réponse** actualise l'état du système de fichiers avec cette écriture et les précédentes. Si la valeur est False, l'**Hôte ECI** peut placer les **Requêtes** d'écriture dans la mémoire tampon (elles risquent malgré tout d'être perdues en cas de défaillance du système).
dataLen: uint	Nombre d'octets à supprimer du fichier. Si cette valeur dépasse la fin du fichier, seuls les octets allant jusqu'à la fin du fichier sont supprimés.

Définitions des paramètres de la réponse:

fileHandle: uchar	Poignée du fichier cible de l'écriture.

Préconditions de la requête:

1) Fichier ouvert en écriture (mode FileWriteOver).

Postconditions de la requête:

1) L'état du fichier sera mis à jour. L'emplacement dans le fichier ne changera pas.
2) Si la suppression et la **synchronisation** aboutissent, les données sont placées dans la mémoire non volatile du système de fichiers de l'**Hôte ECI**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.3.5-1.

Tableau 9.4.5.3.5-1 – Codes d'erreur de la requête resFileWrite

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.6 Message callFileDataLog

C→H callFileDataLog(uchar **fileHandle,** uint **dataLen**, byte **data[])**

- Ce message ajoute des octets dataLen (en données) à la fin du fichier à l'aide d'un tampon système.

Définitions des paramètres de l'appel:

fileHandle: uchar	Poignée du fichier.
dataLen: uint	Nombre d'octets à ajouter à la fin du fichier journal.
data[]: byte	Données à écrire.

Préconditions de l'appel:

1) Fichier ouvert en écriture (mode FileWriteOver ou FileWriteAppend).
2) L'emplacement est fixé à la fin du fichier.
3) Le volume des données à écrire ne provoque pas de problème de quota du système de fichiers.

Postconditions de l'appel:

1) L'état du fichier est mis à jour et l'emplacement dans le fichier passe de "present" à "present+dataLen", sauf en cas d'erreur.
2) Le résultat sera pris en compte dans le système de fichiers de l'**Hôte ECI** sauf en cas de défaillance du système.

Sémantique détaillée:

1) L'**Hôte ECI** placera les données en mémoire tampon et les ajoutera à la fin du fichier dès que possible.

2) Un espace maximal de mémoire tampon alloué à un journal à cette fin est proposé dans le document [b-UIT-T J Suppl. 7].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.3.6-1.

Tableau 9.4.5.3.6-1 – Codes d'erreur de la requête resFileLog

Nom	Description
ErrFileHandleNotExist	Voir le Tableau 9.4.5.5-1 pour la définition.
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	
ErrFileLogNot	

9.4.5.4 Services de répertoire

9.4.5.4.1 Généralités

Les services de répertoire proposent des fonctions de balayage des fichiers des **Clients ECI** disponibles. Les fichiers sont caractérisés par leur nom unique et possèdent des attributs de taille et d'heure/date de la dernière modification. Les messages disponibles sont répertoriés dans le Tableau 9.4.5.4.1-1.

NOTE – L'intégrité de l'attribut d'heure/date est de même niveau que celle du système de fichier et du contenu des fichiers.

Tableau 9.4.5.4.1-1 – Messages des services de répertoire de fichiers

Message	Type	Sens	Étiquette	Description
reqFileStat	A	C→H	0x07	Renvoie la taille et la date/heure de modification du fichier.
reqFileCreate	A	C→H	0x08	Crée un nouveau fichier.
reqFileDelete	A	C→H	0x09	Supprime un fichier.
reqFileDir	A	C→H	0x0A	Répertorie les noms des fichiers disponibles dans le système de fichiers des **Clients ECI**.

9.4.5.4.2 Message reqFileStat

C→H reqFileStat(fileName **filename)** →
H→C resFileStat(uint **size;** long **mtime)**

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** de récupérer la taille et la date/heure de dernière modification d'un fichier stocké.

Définitions des paramètres de la requête:

filename: fileName	Nom du fichier dont les propriétés seront récupérées.

Définitions des paramètres de la réponse:

size: uint	Taille du fichier (en octets).
mtime: long	Heure de la dernière modification synchronisée du fichier.

Préconditions de la requête:

1) Le nom de fichier correspond à un fichier présent dans le système de fichiers.

Postconditions de la requête:

1) Les paramètres **size** et **mtime** correspondent aux propriétés du fichier dont le nom est **filename** ou une erreur s'est produite.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.4.2-1.

Tableau 9.4.5.4.2-1 – Codes d'erreur de la requête resFileStat

Nom	Description
ErrFileNameNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.3 Message reqFileCreate

C→H reqFileCreate(fileName **filename) →**
H→C resFileCreate()

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** de créer un nouveau fichier vide. Un fichier existant de même nom est supprimé.

Définitions des paramètres de la requête:

filename: fileName	Nom du nouveau fichier vide à créer.

Sémantique détaillée:

Le fichier créé survivra à une défaillance du système, sauf en cas de corruption du système de fichiers.

Postconditions de la requête:

1) Le fichier vide nommé filename existe dans le système de fichiers du **Client ECI** et l'horodatage de la modification indique l'heure présente ou le moment où une erreur s'est produite.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.4.3-1.

Tableau 9.4.5.4.3-1 – Codes d'erreur de la requête resFileCreate

Nom	Description
ErrFileQuotaExceeded	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.4 Message reqFileDelete

C→H reqFileDelete(fileName **filename) →**
H→C resFileDelete()

- Ce message supprime un fichier dont le nom est **filename**.

Définitions des paramètres de la requête:

filename: fileName	Nom du nouveau fichier vide à supprimer.

Sémantique détaillée:

- Le fichier créé cessera d'exister après une défaillance du système, sauf en cas de corruption du système de fichiers.

Postconditions de la requête:

1) Le fichier nommé **filename** n'existe pas dans le système de fichiers.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.4.4-1.

Tableau 9.4.5.4.4-1 – Codes d'erreur de la requête resFileDelete

Nom	Description
ErrFileNameNotExist	Voir le Tableau 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.5 Message reqFileDir

C→H reqFileDir(ushort **maxNr) →**
H→C resFileDir(uint **listLen;** fileName **dirList[])**

- Ce message fournit la liste du nombre maximum de noms de fichiers (éléments maxNr). L'ordre de la liste n'est pas défini.

Définitions des paramètres de la requête:

maxNr: ushort	Nombre maximal de noms de fichiers à récupérer.

Définitions des paramètres de la réponse:

listLen: uint	Longueur de la liste en octets.
dirList: filename []	Tableau des noms des fichiers à la disposition du **Client ECI**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.5.4.5-1.

Tableau 9.4.5.4.5-1 – Codes d'erreur de la requête resFileDelete

Nom	Description
ErrFileSystemFailure	Voir le Tableau 9.4.4.7-1.

9.4.5.5 Codes d'erreur de l'API du système de fichiers

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.5.5-1.

Tableau 9.4.5.5-1 – Codes d'erreur de l'API du système de fichiers

Nom	Valeur	Description
ErrFileSystemFailuret	-256	Système de fichiers corrompu ou démonté.
ErrFileNameNotExist	-257	Le nom de fichier n'existe pas dans le système de fichiers.
ErrFileQuotaExceeded	-258	Les ressources du système de fichiers allouées au **Client ECI** ont été dépassées.
ErrFileNameNotExists	-259	Le nom de fichier n'existe pas dans le système de fichiers du **Client ECI**.
ErrFileHandleNotExists	-260	La poignée de fichier n'existe pas (elle a peut-être été fermée précédemment).
ErrFileAppendNot	-261	La tentative d'écriture dans le fichier n'a pas eu lieu à la fin du fichier.
RFU	Autre	Réservé à une utilisation future.

9.4.6 API d'accès à la ressource Temporisateur/Horloge

9.4.6.1 Introduction

Une simple API permet au **Client ECI** d'accéder aux événements de temporisation et à l'heure.

La robustesse de l'horloge doit être définie par un régime de robustesse adapté à toutes les applications d'un **Écosystème ECI**.

- Dans le cas où l'**Écosystème ECI** doit prendre en charge le retour en arrière du système de stockage de fichiers ou des expressions de droits en fonction du temps dans les situations hors ligne, l'horloge doit être robuste afin que les opérations sur la mémoire locale horodatées à partir de cette horloge soient correctement protégées contre toute manipulation.

Le temporisateur permet de générer un message à un moment ultérieur (ajournement). L'événement de temporisation peut être annulé.

NOTE – L'association d'API d'horloge et de temporisateur permet de créer des événements de temporisation périodiques.

Les API du temporisateur et de l'horloge comportent deux parties:

1) API du temporisateur

2) API de l'horloge.

9.4.6.2 API du temporisateur

9.4.6.2.1 Généralités

L'API du temporisateur permet au **Client ECI** de définir un événement de temporisation qui enverra une **Réponse** à l'heure fixée. Si nécessaire, le **Client ECI** peut annuler l'événement. Le nombre d'événements de temporisation en instance à un moment donné peut être limité par les contraintes de mise en oeuvre. Un nombre minimal d'événements de temporisation en instance qu'un **Hôte ECI** prendra en charge pour chaque **Client ECI** est proposé dans le document [b-UIT-T J Suppl. 7]. Les messages de l'API de temporisation sont répertoriés dans le Tableau 9.4.6.2.1-1.

Tableau 9.4.6.2.1-1 – Messages de l'API de temporisation

Message	Type	Sens	Étiquette	Description
reqTimerEvent	A	C→H	0x0	Définit un événement de temporisation futur.
reqTimerCancel	A	C→H	0x1	Annule un événement de temporisation précédemment défini.

9.4.6.2.2 Message reqTimerEvent

C→H reqTimerEvent(uint **timeInterval) →**
H→C resTimerEvent()

- Ce message définit un événement de temporisation futur et reçoit une **Réponse** à l'expiration de la temporisation.

Définitions des paramètres de la requête:

timeInterval: uint	Durée en millisecondes dans l'avenir.

Postconditions de la requête:

- À l'expiration de la valeur en millisecondes de timeInterval, le message resTimerEvent sera envoyé au **Client ECI**, sauf si le message reqTimerCancel est reçu avant.

Préconditions de la réponse:

- L'événement de temporisation est arrivé à expiration et aucun message reqTimerCancel n'a été reçu.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.6.2.2-1.

Tableau 9.4.6.2.2-1 – Codes d'erreur de la requête resTimerEvent

Nom	Description
ErrTimerMaxExceeded	Voir le Tableau 9.4.6.4-1.

9.4.6.2.3 Message reqTimerCancel

C→H reqTimerCancel(msgId **id) →**
H→C resTimerCancel()

- Ce message annule l'événement de temporisation précédemment défini en fonction de l'identificateur de message de la **Requête** d'origine.

Définitions des paramètres de la requête:

id: msgId	Annule l'événement de temporisation défini par un message asynchrone comportant l'identificateur du message.

Préconditions de la requête:

1) L'identificateur a été renvoyé du fait du message reqTimerEvent et un événement de temporisation n'est pas encore parvenu à expiration.

Postconditions de la réponse:

1) L'événement de temporisation est annulé – Aucun message resTimerCancel ne sera envoyé ou une erreur est renvoyée.
2) Des erreurs TimerExpired se produiront si l'événement de temporisation a été annulé mais que le message **resTimerEvent** a été reçu avant le message **resTimerCancel.**

9.4.6.3 API d'horloge

9.4.6.3.1 Généralités

L'API d'horloge permet au **Client ECI** de lire l'horloge sous forme de nombre entier et de convertir celui-ci à l'heure locale. Les messages de l'API d'horloge sont répertoriés dans le Tableau 9.4.6.3.1-1.

Tableau 9.4.6.3.1-1 – Messages de l'API d'horloge

Message	Type	Sens	Étiquette	Description
getTime	S	C→H	0x3	Lit l'horloge système locale sous forme de nombre entier.
callLocaltime	S	C→H	0x4	Convertit le nombre entier en heure locale.

9.4.6.3.2 Message getTime

C→H long **getTime()**

- Ce message renvoie l'heure en secondes à partir du 1er janvier 1970, 0:00 GMT.

9.4.6.3.3 Message callLocaltime

C→H callLocaltime(long **time;** tm ***tim)**

- Ce message convertit la valeur de **time** en représentation humaine et est défini dans la structure **tim**. Analogue
  à la fonction de bibliothèque C localtime provenant de <time.h>.

Définitions des paramètres de l'appel:

time: long	Heure sous la forme d'un nombre entier représentant des secondes à partir du 1er janvier 1970, 0:00 GMT, à convertir en heure locale.
tim: tm *	Pointeur vers la structure **tm** qui sera fixée à l'heure locale et est définie dans le Tableau 9.4.6.3.3-1.

Tableau 9.4.6.3.3-1 – Définition du type de la structure tm représentant l'heure humaine

```
typedef struct tm {
   int tm_sec;  // 0 .. 59 secondes ou 60 en cas de seconde intercalaire
   int tm_min;  // 0 .. 59 (minutes)
   int tm_hour     // 0 .. 23 (heures)
   int tm_mday; // 1 .. 31 (jours dans le mois)
   int tm_mon;     // 1 .. 12 (mois)
   int tm_year; // année - 1900
   int tm_wday; // 0 .. 6 (jours de la semaine; 0=dimanche)
   int tm_yday; // 0 .. 365 (jour de l'année; 0=1er janvier)
   int tm_isdst;   // 1=application de l'heure d'été, 0=pas d'heure d'été
   char tm_zone[15]; // chaîne du fuseau horaire, p. ex., GMT, CET
```

```
    int tm_gmtoff;  // décalage local par rapport à l'heure GMT
} tm ;
```

9.4.6.4 Codes d'erreur des API de temporisateur et d'horloge

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.6.4-1.

Tableau 9.4.6.4-1 – Codes d'erreur des API de temporisateur et d'horloge

Nom	Valeur	Description
ErrTimerMaxExceeded	256	Durée maximale de temporisation dépassée.
RFU	autre	Réservé à une utilisation future.

9.4.7 API d'accès à la gestion de la consommation d'énergie

9.4.7.1 Introduction

Le **Client ECI** peut accéder à l'interface de gestion de la consommation d'énergie de l'**Hôte ECI**. Cette interface permet au **Client ECI** de réduire la puissance consommée, en mode simple ou négocié, en cas de mise en veille du système. Elle lui permet également de se relancer et de réveiller ultérieurement l'**Équipement CPE** afin d'exécuter des fonctions en arrière-plan. Les différents états de consommation d'énergie de l'**Hôte ECI** sont les suivants:

- **PwrOn:** l'**Hôte ECI** est fonctionnel et ne prévoit pas de réduction de puissance.
- **PwrToStby:** l'**Hôte ECI** a l'intention de passer en mode veille (mais peut revenir à l'état PwrOn). En général, tous les **Clients ECI** doivent réduire leur consommation d'énergie.
- Mode veille: l'**Hôte ECI** et le **Client ECI** ne sont pas fonctionnels. L'**Équipement CPE** (et donc l'**Hôte ECI** et le **Client ECI**) peuvent se réveiller de cet état en cas d'événements fixés à l'avance (en général, une temporisation).
- Power-off: l'**Équipement CPE** est hors tension. L'**Hôte ECI** et le **Client ECI** ne fonctionnent pas.

Les **Clients ECI** peuvent fonctionner en mode gestion de consommation d'énergie simple et être arrêtés si et quand l'**Hôte ECI** l'estime nécessaire. Alternativement, ils peuvent envoyer un message **PwrInfo(PwrInfoOn)** pour demander à passer en mode géré. Dans ce dernier cas, l'**Hôte ECI** les informera de son intention de réduire la consommation d'énergie à l'aide du message **reqPwrChange**, pour lequel le **Client ECI** pourra accuser réception avec un message **resPwrChange(PwrDown).** Il pourra également repousser la mise en veille à l'aide d'un paramètre approprié du message **resPwrChange(PwrUp)** jusqu'à ce qu'il ait terminé et soit prêt à changer de mode. L'**Hôte ECI** réitérera régulièrement le message **reqPwrChange**.

NOTE – Il n'est pas possible de garantir totalement que le **Client ECI** sera systématiquement en mesure de terminer toutes ses activités (par exemple, en cas de panne de courant incontrôlée ou d'ajournement prolongé du passage en mode veille).

La Figure 9.4.7.1-1 présente les états de l'**Hôte ECI** et les conditions de leurs changements ainsi que les actions/messages déclenchés lors du passage à des **Clients ECI** en mode géré.

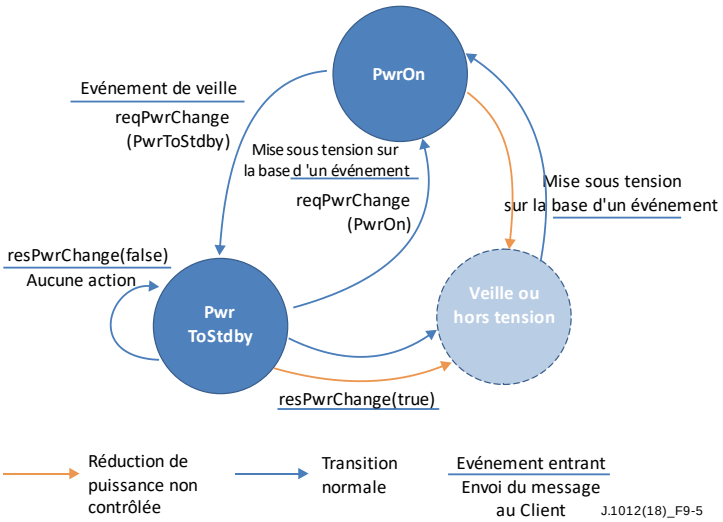


Figure 9.4.7.1-1 – Modes de consommation d'énergie de l'Hôte ECI et principales interactions avec un Client géré

Les **Clients ECI** et les **Hôtes ECI** seront en mesure de gérer le rétablissement après un *événement de réduction de consommation d'énergie incontrôlé*. Dans ce cas, il est autorisé de bloquer temporairement les fonctionnalités du **Client ECI** et de l'**Hôte ECI** afin de tenter de limiter les problèmes pour l'**Utilisateur**.

Les **Équipements CPE** peuvent être dotés d'options de réveil à partir d'une réduction de puissance en cas d'événements réseau ou d'autres modes à consommation d'énergie réduite. L'interface **ECI** ne définit pas de comportement spécifique pour ce type de mode de consommation d'énergie et leur interaction avec l'**Hôte ECI** ou les **Clients ECI**, si ce n'est que leurs services respectifs continueront à fonctionner si l'**Hôte ECI** se trouve en mode **PwrOn** ou **PwrToStdby**. Notamment, il n'existe pas d'état spécifique à la suspension de l'exécution.

Les **Clients ECI** pourront demander à l'**Hôte ECI** de les réveiller à une certaine heure à l'avenir et de leur envoyer un message.

L'API de gestion de la consommation d'énergie comporte les groupes de messages suivants:

1) Transitions de consommation d'énergie: gestion de l'arrêt ordonné des **Clients ECI**. Les détails sont définis dans le § 9.4.7.2.

2) Fonctions de rétablissement temporisé de la consommation d'énergie pour le compte des **Clients ECI** (détails définis au § 9.4.7.3).

9.4.7.2 Définition des messages de l'API de gestion des transitions de consommation d'énergie

9.4.7.2.1 Généralités

Ce paragraphe concernant l'API de gestion de la consommation d'énergie définit les fonctionnalités permettant aux **Clients ECI** de s'arrêter en réponse à un événement de réduction de puissance annoncé de l'**Hôte ECI** de façon à fournir un service optimal à l'**Utilisateur**. Les messages définis sont répertoriés dans le Tableau 9.4.7.2.1-1.

Tableau 9.4.7.2.1-1 – Messages relatifs aux transitions de consommation d'énergie

Message	Type	Sens	Étiquette	Description
getPwrStatus	S	C→H	0x0	Obtient la valeur actuelle de l'état de consommation d'énergie.
setPwrInfo	S	C→H	0x1	Demande la notification des changements d'état de consommation d'énergie.
reqPwrChange	A	H→C	0x2	Notification d'un changement d'état de consommation d'énergie.

Les **Clients ECI** ne s'arrêteront pas après l'envoi d'un message **resPwrInfo(PwrDown)** mais seront prêts à reprendre leur fonctionnement normal sur réception d'un message **reqPwrChange(PwrOn)**.

9.4.7.2.2 Message getPwrStatus

C→H uchar **getPwrStatus()**

- Ce message renvoie l'état de consommation d'énergie actuel de l'**Hôte ECI**.

Définition de la propriété: voir le Tableau 9.4.7.2.2-1.

Tableau 9.4.7.2.2-1 – Valeurs de l'état de consommation d'énergie de l'Hôte

Nom	Valeur	Description
PwrOn	0x00	Adresse IP par défaut de l'**Hôte ECI**.
PwrToStdby	0x01	Adresse IP de l'**Hôte ECI** utilisée pour une communication WAN (Internet).
RFU	autre	Réservé à une utilisation future.

9.4.7.2.3 Message setPwrInfo

C→H setPwrInfo(bool **pwrInfo)**

- Ce message permet de passer en mode consommation d'énergie gérée et de le quitter ainsi que de contrôler l'**Hôte ECI** envoyant au **Client ECI** les messages **resPwrChange** lors d'événements de changement d'état de consommation d'énergie.

Définition de la propriété:

- **pwrInfo** égale **true** est le mode consommation d'énergie gérée. **pwrInfo** égale **false** est le mode consommation d'énergie non gérée.

Description sémantique:

- Lorsque la valeur de **pwrInfo** est **True**, l'**Hôte ECI i**nforme le **Client ECI** des changements d'état de consommation d'énergie et ne l'arrête que lorsque ce client a confirmé la réception d'un message reqPwrChange(PwrToStby). Lorsque la valeur de **pwrInfo** est **False**, l'**Hôte ECI** n'informe pas le **Client ECI** des changements d'état de consommation d'énergie et réduit la consommation d'énergie du Client à son gré.
- Au démarrage, la valeur de **PowerInfo** de chaque **Client ECI** est **False**.

NOTE – Il est suggéré aux **Clients ECI** fonctionnant en mode réduction de consommation d'énergie gérée de ne pas commencer d'activités sensibles à ce mode de consommation avant d'avoir envoyé le message **reqPwrInfo(True)** à l'**Hôte ECI**.

9.4.7.2.4 Message reqPwrChange

H → C reqPwrChange(uchar **hostPwrState)** **→**
C→H resPowerChange(bool **ready)**

- Ce message signale un changement d'état de consommation d'énergie et si l'argument est **PwrToStdby Requests**, le **Client ECI** peut soit en accuser réception et passer en mode veille de façon contrôlée, soit refuser s'il est en train d'effectuer des tâches logicielles importantes.

Définitions des paramètres de la requête:

hostPwrState: uchar	Nouvel état de consommation d'énergie de l'**Hôte ECI**. Les valeurs possibles sont définies dans le Tableau 9.4.7.2.2-1.

Définitions des paramètres de la réponse:

ready: bool	Indique que le **Client ECI** est prêt à passer en mode veille.

Description sémantique:

- L'**Hôte ECI** renverra ce message si la **Réponse** du **Client ECI** est négative (non prêt). Des chiffres relatifs au taux minimum de répétition et au délai d'attente sont proposés dans le document [b-UIT-T J Suppl. 7].

Préconditions de la requête:

1) PwrInfo == True.
2) Un changement (récent) d'état de consommation d'énergie chez l'**Hôte ECI** a eu lieu et le **Client ECI** n'a pas (encore) confirmé qu'il était prêt à passer en mode veille.

Postconditions de la réponse:

1) Le **Client ECI** est prêt à passer en mode veille si **ready == True**; il ne l'est pas si **ready == False**.

Les codes d'erreur sont définis dans le Tableau 9.4.7.2.4-1.

Tableau 9.4.7.2.4-1 – Codes d'erreur de la requête ansPwrChange

Nom	Description
ErrPwrInfoNot	Voir le Tableau 9.4.7.4-1.

NOTE – Pour les **Hôtes ECI**, l'erreur **errPwrInfoNot** n'a qu'un but informatif.

9.4.7.3 Définition des messages relatifs à la sortie du mode veille

9.4.7.3.1 Généralités

Ce paragraphe concernant l'API de gestion de la consommation d'énergie définit les fonctionnalités permettant aux **Clients ECI** de reprendre leur exécution à une heure préprogrammée, en réveillant l'**Équipement CPE** si nécessaire. Les messages définis sont répertoriés dans le Tableau 9.4.7.3.1-1.

Tableau 9.4.7.3.1-1 – Messages relatifs à la sortie du mode veille

Message	Type	Sens	Étiquette	Description
setPwrWakeup	set	C→H	0x3	Définit l'heure du réveil du **Client ECI**.
reqPwrWakeupEvent	A	H→C	0x4	Signale l'expiration de la temporisation de réveil.

9.4.7.3.2 Message setPwrWakeup

C→H setPwrWakeup(uint **time)**

- Ce message définit un événement de temporisation: après l'heure **time**, l'**Hôte ECI** réveillera le **Client ECI** si nécessaire et enverra un message **reqPwrWakeupEvent()**.

Définition de la propriété:

time: uint	Durée en secondes avant que l'**Hôte ECI** génère un événement de réveil pour le **Client ECI**. La valeur 0 signifie que le **Client ECI** ne requiert pas d'événement de réveil.

Sémantique détaillée:

- Si l'**Hôte ECI** le peut, il sortira du mode veille et démarrera le **Client ECI immédiatement**. Dans le cas contraire, il enverra l'événement de réveil dès que possible. Des exigences relatives à la précision de la durée sont proposées dans le document [b-UIT-T J Suppl. 7].

9.4.7.3.3 Message reqPwrWakeupEvent

H→C reqPwrWakeupEvent() →
C→H resWakeupEvent()

- Ce message notifie le **Client ECI** de l'expiration de la temporisation de réveil. Le **Client ECI** accuse réception de cette **Requête** avec une **Réponse** à la fin du traitement critique de l'événement de réveil.

Sémantique détaillée:

- L'**Hôte ECI** tentera de renvoyer ce message lors d'initialisations successives du **Client ECI** jusqu'à ce que ce dernier en accuse réception par un message **PwrWakeupEvent()**. L'événement est envoyé en mode **PwrOn** mais repoussé en mode **PwrToStdby**.

Préconditions de la requête:

1) Un événement de temporisation de réveil pour le **Client ECI** a été défini antérieurement et a expiré.
2) L'événement n'a pas encore fait l'objet d'une **Réponse**.
3) L'**Hôte ECI** est en mode **PwrOn.**

Postconditions de la réponse:

1) L'**Hôte ECI** cessera d'envoyer des messages **reqPwrWakeupEvent()** basés sur l'événement de changement de mode de consommation d'énergie de la **Requête** correspondante (voir la **précondition 2**).

9.4.7.4 Codes d'erreur de l'API des transitions de consommation d'énergie

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.7.4-1 ci-dessous.

Tableau 9.4.7.4-1 – Codes d'erreur de l'API des transitions de consommation d'énergie

Nom	Valeur	Description
ErrPwrInfoNot	-256	Le **Client ECI** indique qu'il n'a pas demandé à être informé des changements d'état de consommation d'énergie.

9.4.8 API d'accès à la ressource de définition des paramètres de pays et de langue

9.4.8.1 Introduction

L'API de définition des paramètres de pays et de langue permet aux **Clients ECI** et aux **Hôtes ECI** de les demander à l'**Utilisateur** à partir de l'**Hôte ECI** ou d'un **Client ECI** respectivement. Les messages relatifs à l'API de définition des paramètres de pays et de langue sont répertoriés dans le Tableau 9.4.8.1-1.

Tableau 9.4.8.1-1 – Messages relatifs à l'API de définition des paramètres de pays et de langue

Message	Type	Sens	Étiquette	Description
reqHCountry	A	C→H	0x0	Demande la préférence de pays existante de l'**Hôte ECI**.
reqCCountry	A	H→C	0x1	Demande la préférence de pays existante du **Client ECI**.
reqHLanguage	A	C→H	0x2	Demande la préférence de langue existante de l'**Hôte ECI**.
reqCLanguage	A	H→C	0x3	Demande la préférence de langue existante du **Client ECI**.

9.4.8.2 Définitions des messages de l'API des paramètres de pays et de langue

9.4.8.2.1 Message reqHCountry setting

C→H reqHCountry() →
H→C resHCountry setting (uint **iso_3166_country_code)**

- Ce message permet au **Client ECI** de demander le paramètre du pays de résidence de l'**Utilisateur** et de recevoir une **Réponse** de l'**Hôte ECI** contenant le paramètre de pays stocké.

Définitions des paramètres de la réponse:

iso_3166_country_code: uint	Ce champ contient le paramètre de pays actuel de l'**Hôte ECI**. Le code de pays est un champ de 24 bits identifiant le pays de l'Hôte à l'aide de trois caractères en majuscules comme spécifié par la norme ISO 3166-1 alpha 3 [ISO 3166-1]. Chaque caractère est codé à 8 bits conformément à [ISO/CEI 8859-1].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.8.2.1-1.

Tableau 9.4.8.2.1-1 – Codes d'erreur de la requête reqHCountry

Nom	Description
ErrCountryNotExists	Voir le Tableau 9.4.8.2.5-1.

9.4.8.2.2 Message reqCCountry setting

H→C reqCCountry() →
C→H resCCountry setting (uint **iso_3166_country_code)**

- Ce message permet à l'**Hôte ECI** de demander le paramètre du pays de résidence de l'**Utilisateur** et de recevoir une **Réponse** du **Client ECI** contenant le paramètre de pays stocké.

Définitions des paramètres de la réponse:

iso_3166_country_code: uint	Ce champ contient le paramètre de pays actuel de l'**Hôte ECI**. Le code de pays est un champ de 24 bits identifiant le pays de l'Hôte à l'aide de trois caractères en majuscules comme spécifié par la norme ISO 3166-1 alpha 3 [ISO 3166-1]. Chaque caractère est codé à 8 bits conformément à [ISO/CEI 8859-1].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.8.2.2-1.

Tableau 9.4.8.2.2-1 – Codes d'erreur de la requête reqCCountry

Nom	Description
ErrCountryNotExists	Voir le Tableau 9.4.8.2.5-1.

9.4.8.2.3 Message reqHLanguage setting

H→C reqHLanguage(uint **iso_3166_language_code) →**
C→H resHLanguage setting()

- Ce message permet au **Client ECI** de demander le paramètre réel du choix de langue de l'**Utilisateur** et de recevoir une **Réponse** de l'**Hôte ECI** contenant le paramètre de langue stocké.

Définitions des paramètres de la réponse:

iso_3166_language_code: uint	Ce champ contient le paramètre de préférence de langue de l'**Hôte ECI**. Ce champ de 24 bits identifie la langue à l'aide de trois caractères en minuscules comme spécifié dans [ISO 639-2]. Les normes ISO 639-2/B et ISO 639-2/T peuvent être toutes deux utilisées. Chaque caractère est codé à 8 bits conformément à [ISO/CEI 8859-1].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.8.2.3-1.

Tableau 9.4.8.2.3-1 – Codes d'erreur de la requête reqHLanguage

Nom	Description
ErrLanguageNotExists	Voir le Tableau 9.4.8.2.5-1.

9.4.8.2.4 Message reqCLanguage setting

H→C reqCLanguage(uint **iso_3166_language_code) →**
C→H resCLanguage setting()

- Ce message permet à l'**Hôte ECI** de demander la préférence de langue de l'**Utilisateur** et de recevoir une **Réponse** du **Client ECI** contenant le paramètre de langue stocké.

Définitions des paramètres de la réponse:

iso_3166_language_code: uint	Ce champ contient le paramètre de préférence de langue de l'**Hôte ECI**. Ce champ de 24 bits identifie la langue à l'aide de trois caractères en minuscules comme spécifié dans [ISO 639-2]. Les normes ISO 639-2/B et ISO 639-2/T peuvent éventuellement être toutes deux utilisées. Chaque caractère est codé à 8 bits conformément à [ISO/CEI 8859-1].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.4.8.2.4-1.

Tableau 9.4.8.2.4-1 – Codes d'erreur de la requête reqCLanguage

Nom	Description
ErrLangageNotExists	Voir le Tableau 9.4.8.2.5-1.

9.4.8.2.5 Codes d'erreur de l'API de définition des paramètres de pays et de langue

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.4.8.2.5-1 ci-dessous.

Tableau 9.4.8.2.5-1 – Codes d'erreur de l'API de définition des paramètres de pays et de langue

Nom	Valeur	Description
ErrCountryNotExists	-256	L'**Hôte ECI** indique que l'**Utilisateur** n'a pas encore déclaré son pays de résidence.
ErrLangageNotExists	-257	L'**Hôte ECI** indique que l'**Utilisateur** n'a pas encore déclaré sa préférence de langue pour la communication avec l'interface d'utilisateur.

9.5 API relatives aux ressources spécifiques à l'interface ECI de l'Hôte ECI

9.5.1 Liste des API relatives aux ressources spécifiques à l'interface ECI de l'Hôte ECI

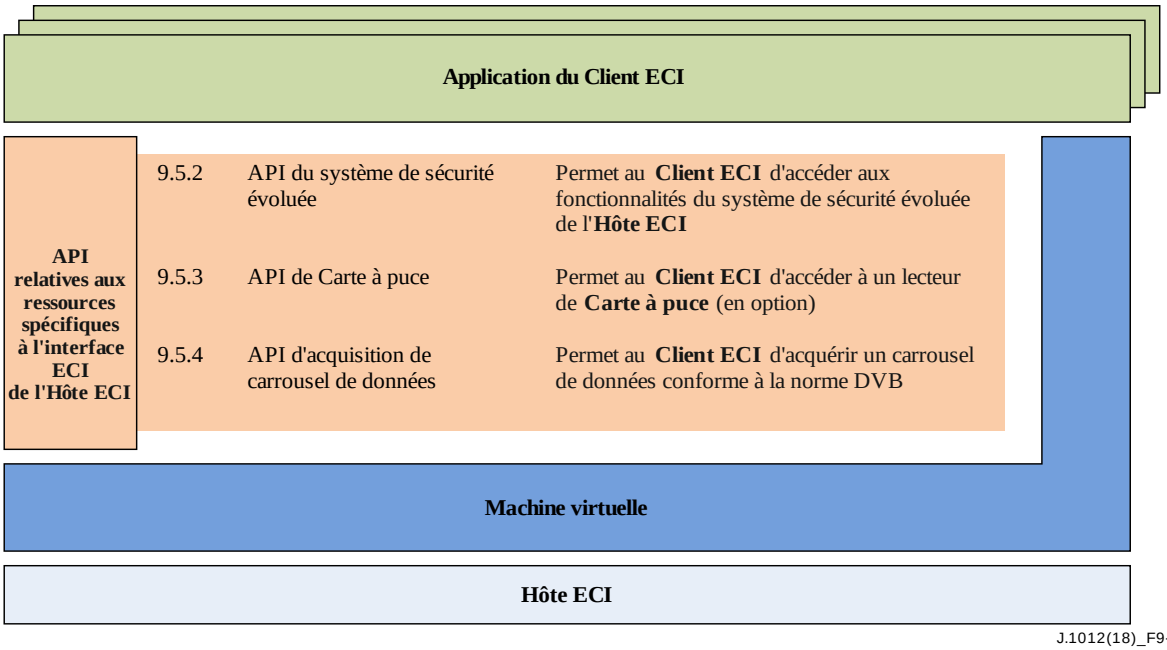


Figure 9.5.1-1 – Représentation des API définies au § 9.5

Le Tableau 9.5.1-1 répertorie les API présentées dans le § 9.8 et le Tableau 9.5.1-1 illustre le positionnement des API définies au § 9.5 dans l'**architecture ECI**.

Tableau 9.5.1-1 – Liste des API définies au § 9.5

Paragraphe	Nom de l'API	Description
9.5.2	API du système de sécurité évoluée	Permet au **Client ECI** d'accéder aux fonctionnalités du système de sécurité évoluée de l'**Hôte ECI**.
9.5.3	API de **Carte à puce**	Permet au **Client ECI** d'accéder à un lecteur de **Carte à puce** (en option).
9.5.4	API d'acquisition de carrousel de données	Permet au **Client ECI** d'acquérir un carrousel de données conforme à la norme DVB.

9.5.2 API du système de sécurité évoluée

9.5.2.1 Introduction

Lors du chargement d'un **Client ECI**, l'**Hôte ECI** lui alloue un créneau de sécurité évoluée adapté (un type de **Client ECI** ou de **Micro serveur**). Ce créneau sera disponible pendant le cycle de vie du **Client ECI** concerné. Pour initialiser le créneau, l'**Hôte ECI** chargera la **Chaîne de certificats d'opérations de plate-forme** contenant la clé publique d'**Opération de plate-forme.** Cette action lie les échanges ultérieurs avec le **Créneau AS** (sécurité évoluée) au détenteur de la clé secrète d'**Opération de plate-forme**.

L'API du système de sécurité évoluée permet aux **Clients ECI** d'interagir avec la fonction de sécurité évoluée de l'**Équipement CPE**. Les divers types d'échanges possibles entre la fonction de sécurité évoluée et les **Clients ECI** sont en général lancés par ces derniers. Le **Client ECI** reçoit un signal à la fin de l'exécution des opérations de sécurité évoluée longues.

Le **Créneau AS** prenant en charge plusieurs sessions, il est possible de réutiliser les informations stockées (état et configuration) dans le Créneau de sécurité évoluée lors de multiples sessions de déchiffrement et de **Rechiffrement** des médias. Le **Créneau AS** stocke une clé intermédiaire dite

"clé de liaison" (LK_1) de niveau supérieur par session. Il est possible de calculer rapidement les nouveaux mots de contrôle des sessions à partir de leur clé LK_1.

Le **Créneau AS** peut également calculer une "clé d'authentification" secrète utilisable avec les applications des **Clients ECI**, qui renforce la sécurité de l'envoi d'informations secrètes aux **Clients ECI**.

La configuration du **Créneau AS** est initialisée par le **Client ECI** et définit son mode opératoire. Le **Créneau AS** permet au client d'authentifier sa configuration. Il existe deux modes majeurs d'authentification:

1) **Mode Echelle de clés.** Authentification dans le cadre du calcul des mots de contrôle: la configuration du créneau a été utilisée dans le calcul pour générer le mot de contrôle qui a chiffré le contenu. La même information est requise pour calculer le mot de contrôle correct permettant de déchiffrer le contenu, ce qui authentifie implicitement la configuration.
2) **Mode Clé d'authentification.** L'authentification résulte d'une fonction de validation explicite utilisant des données de vérification qui ne peuvent être générées que par le fournisseur du **Client ECI**. Cette fonction est pratiquement obligatoire pour les **Créneaux AS** configurés pour le rechiffrement car un déchiffrement correct ne peut pas servir de moyen de vérification de cette opération.

En sus des modes ci-dessus, le **Client ECI** peut requérir l'exécution d'une revérification lors de l'initialisation de chaque créneau en exigeant une "authentification en ligne". Alternativement, une "authentification hors ligne" peut être effectuée. Pour que l'authentification réussisse, il faut que le mode sélectionné corresponde aux données d'authentification communiquées par le fournisseur.

L'API globale du système de sécurité évoluée est fractionnée en plusieurs API distinctes permettant de tenir compte des capacités des **Hôtes ECI** et des **Clients ECI** qui l'utilisent:

1) *API générale du système de sécurité évoluée:* définit les fonctionnalités génériques du système de sécurité évoluée. Tous les **Hôtes ECI** et les **Clients ECI** la prendront en charge.
2) *API de déchiffrement du système de sécurité évoluée*: définit la fonctionnalité de déchiffrement du système de sécurité évoluée. Tous les **Hôtes ECI** et les **Clients ECI** capables de procéder au déchiffrement la prendront en charge.
3) *API d'exportation du système de sécurité évoluée:* définit les fonctionnalités d'exportation du système de sécurité évoluée. Tous les **Hôtes ECI** et les **Clients ECI** capables de procéder au déchiffrement et à l'exportation la prendront en charge. Les **Hôtes ECI** capables de procéder à l'exportation prendront également en charge le chiffrement.
4) *API de chiffrement du système de sécurité évoluée*: définit la fonctionnalité de chiffrement du système de sécurité évoluée. Tous les **Hôtes ECI** et les **Clients ECI** capables de procéder au chiffrement la prendront en charge.

La contrainte suivante s'appliquera:

- Le **Client ECI** prendra en charge soit le déchiffrement, soit le chiffrement, mais pas les deux simultanément.

L'**Hôte ECI** et le **Client ECI** utiliseront la ressource de découverte des interfaces de l'**Hôte ECI** pour échanger des informations sur leurs capacités respectives. L'**Hôte ECI** attribuera le créneau approprié en fonction du résultat de la découverte: un créneau de chiffrement pour les **Clients ECI** nécessitant un chiffrement, et un Créneau AS de déchiffrement pour les **Clients ECI** nécessitant un déchiffrement.

NOTE – Les fonctions assurant des fonctionnalités complémentaires peuvent exister dans différentes API – l'API AS générale et une API AS plus spécifique.

Les messages de l'API AS générale ne doivent être pris en charge par l'**Hôte ECI** que si cela est nécessaire pour refléter les capacités de l'**Hôte ECI** (prise en charge du déchiffrement, de l'exportation et du chiffrement).

Les messages des API AS sont définis conformément aux fonctions de sécurité évoluée décrites aux § 8.2.4 et 9.9 de la Recommandation [UIT-T J.1014]. Le § 8.2.4.1 y fournit un aperçu des fonctions. Le premier paramètre, slotId, n'est pas inclus dans les définitions de la Recommandation [UIT-T J.1014]: il est fourni par l'**Hôte ECI**.

Nombre des définitions de type et de valeur des paramètres utilisés dans cette définition d'API sont décrites dans la Recommandation [UIT-T J.1014]. Les codes d'erreur de cette API y sont également définis et ne sont pas répertoriés spécifiquement dans la présente Recommandation message par message. Les codes d'erreur des valeurs des paramètres correspondent au décompte de séquence de paramètres comme défini dans les fonctions de la Recommandation [UIT-T J.1014] indiquées en référence, qui comportent généralement un paramètre supplémentaire (slotId).

9.5.2.2 Définitions des messages de l'API AS générale

9.5.2.2.1 Généralités

L'API AS générale fournit les messages répertoriés dans le Tableau 9.5.2.2.1-1.

Tableau 9.5.2.2.1-1 – Messages de l'API AS générale

Message	Type	Sens	Étiquette	Description
reqAsInitSlot	A	C→H	0x0	Initialise le **Créneau AS**.
callAsNextKeySession	S	C→H	0x1	Passer à la clé aléatoire suivante pour une session.
reqAsStopSession	A	C→H	0x2	Arrêter une session.
reqAsLoadSlotLk	A	C→H	0x3	Calculer la clé de liaison de niveau supérieur (LK1).
reqAsComputeAkClient	A	C→H	0x4	Calculer la clé d'authentification pour les applications du **Client ECI**.
reqAsClientChalResp	A	C→H	0x5	Appliquer la clé d'authentification du **Client ECI** aux données et renvoyer un résultat.
getAsSlotRk	S	C→H	0x6	Obtenir une valeur de clé aléatoire pour le **Créneau AS**.
getAsSessionRk	S	C→H	0x7	Obtenir une valeur de clé aléatoire pour une session.
getAsSessionLimitCounter	S	C→H	0x8	Obtenir la valeur du compteur de limite pour la session.
setAsSessionLimitEvent	S	C→H	0x9	Définir la valeur limite pour l'envoi d'un message reqAsEventSessionLimit au **Client ECI**.
reqAsEventSessionLimit	A	H→C	0xA	Lorsqu'une valeur limite est atteinte pour les unités restantes, envoyer l'événement au **Client ECI**.
getAsClientRnd	S	C→H	0xB	Obtenir un nouveau nombre aléatoire pour les applications du **Client ECI**.
getAsSC	S	C→H	0xC	Obtenir le statut du champ de commande d'embrouillage actuel du contenu d'une session.
reqAsEventSC	A	H→C	0xD	Message d'événement lors du changement du champ de commande d'embrouillage dans la session.
getChipsetId	S	C→H	0xE	Obtenir la valeur ChipsetID du **Bloc d'échelle de clés**
getImageTargetId	S	C→H	0xF	Obtenir la valeur ECI_Image_Target_Id de l'équipement CPE

9.5.2.2.2 Message reqAsInitSlot

C→H reqAsInitSlot(uint **slotVersion,** uint **slotMode →**
H→C resAsInitSlot()

- Ce message initialise le créneau avec divers paramètres généraux.

Définitions des paramètres de la requête:

slotVersion: uint	Version de la fonctionnalité du créneau telle que définie dans la Recommandation [UIT-T J.1014]
slotMode: uint	Mode de fonctionnement principal pour le créneau, voir la Recommandation [UIT-T J.1014].

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsInitSlot définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur des paramètres slotId et POPKchain.

9.5.2.2.3 Message callAsNextKeySession

C→H callAsNextKeySession(uint **sessionId**)

- Ce message entraîne un passage à la clé aléatoire suivante pour une session.

Définitions des paramètres de la requête:

sessionId: uint	Session pour laquelle un passage à la clé aléatoire suivante est annoncé.

Description sémantique:

- Ce message est équivalent au message de sécurité évoluée callAsNextKeySession défini dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.4 Message reqAsStopSession

C→H reqAsStopSession(uint **sessionId**) →
H→C resAsStopSession()

- Ce message met fin à une session de **Créneau AS**.

Définitions des paramètres de la requête:

sessionId: uint	Identificateur de la session à arrêter.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsStopSession définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.5 Message reqAsLoadSlotLk

C→H reqAsLoadSlotLk(uint **sessId**, InputV **inputV**, ulong **spkUri,** uchar **spkIndx**) →
H→C resAsLoadSlotLk()

- Ce message calcule la clé de liaison de niveau supérieur LK_1 qui peut par la suite être utilisée pour calculer les mots de contrôle.

Définitions des paramètres de la requête:

sessId: uint	Identificateur de la session à initialiser.
inputV: InputV	Message contenant la clé de liaison LK_1 chiffrée avec la clé publique du jeu de puces et protégée par la signature à clé secrète de l'émetteur.
spkUri: ulong	Règles d'utilisation du vecteur de Clés publiques de l'émetteur, utilisé par la suite pour calculer un mot de contrôle, voir la Recommandation [UIT-T J.1014].
spkIndx: uchar	Index définissant l'emplacement de la Clé publique d'émetteur du Créneau AS dans le vecteur de Clés publiques d'émetteur et utilisé par la suite pour calculer un mot de contrôle, voir le § 7 de la Recommandation [UIT-T J.1014].

Description sémantique:

- Ce message est équivalent à la fonction de **Créneau AS** reqAsLoadLk1 définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

- L'**Hôte ECI** émettra également une fonction reqAsDecoupleDecryptSession (Recommandation [UIT-T J.1014]) si une session de déchiffrement de **Créneau AS** précédemment associée à une autre session de déchiffrement de **Créneau AS** est arrêtée (voir § 9.5.2.3.1).

9.5.2.2.6 Message reqAsComputeAkClient

C→H reqAsComputeAkClient(InputV **inputV,** uint **nSpk** uchar **spkIndx,** PubKey **spk[16],** PubKey **popk[16],** SessionConfig **akCnf[16],** ulong **spkUri;** uchar **XT[32],** bool **online)** → **H→C resAsComputeAkClient** ()

- Ce message calcule une clé d'authentification pour utilisation par le **Client ECI**.

Définitions des paramètres de la requête:

inputV: InputV	Message contenant la valeur r chiffrée avec la clé publique du jeu de puces et protégée par la signature à clé secrète de l'émetteur, laquelle valeur est utilisée pour calculer la clé d'authentification.
nSpk: uint	Nombre de valeurs dans le vecteur de Clés publiques de l'émetteur, voir la Recommandation [UIT-T J.1014].
spkIndx: uchar	Index définissant l'emplacement de la Clé publique d'émetteur du Créneau AS dans le vecteur de Clés publiques de l'émetteur, la valeur de la Clé publique d'opération de plate-forme du Créneau AS dans le vecteur de Clés publiques d'opération de plate-forme et le champ slotConfig du vecteur clCnf utilisé par la suite pour calculer la clé d'authentification du client, voir la Recommandation [UIT-T J.1014].
spk[16]: PubKey	Vecteur de **Clés publiques de l'émetteur,** utilisé pour calculer la clé d'authentification du client; voir la Recommandation [UIT-T J.1014].
popk[16]: PubKey	Vecteur de Clés publiques d'opérateurs de plate-forme, utilisé pour calculer la clé d'authentification du client; voir la Recommandation [UIT-T J.1014].
akCnf[16]: SessionConfig	Vecteur de configurations de session de client, utilisé pour calculer la clé d'authentification du client; voir la Recommandation [UIT-T J.1014].
spkUri: ulong	Règles d'utilisation du vecteur de Clés publiques de l'émetteur, utilisé par la suite pour calculer un mot de contrôle, voir la Recommandation [UIT-T J.1014].
XT[32]: uchar	Valeur du champ d'extension utilisé pour calculer la clé d'authentification du client; voir la Recommandation [UIT-T J.1014]. La valeur par défaut est { 0x00 }.
online: bool	Si la valeur est "true", la clé aléatoire du créneau est utilisée pour le calcul de la clé d'authentification, obligeant le fournisseur à procéder à un nouveau calcul de clé d'authentification.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsComputeAkClient définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.7 Message reqAsClientChalResp

C→H reqAsClientChalResp(uchar **challenge**[16]);→
H→C reqAsClientChalResp(uchar **response**[16])

- Ce message utilise la clé d'authentification du client calculée par le message reqAsComputeAkClient (défini dans la Recommandation [UIT-T J.1014]) pour déchiffrer en entrée un paramètre challenge et produire en sortie un paramètre response, chacun composé de 128 bits.

Définitions des paramètres de la requête:

challenge[16]: uchar	Entrée de 128 bits devant être déchiffrée par la clé d'authentification du client.

Définitions des paramètres de la réponse:

response[16]: uchar	Sortie de 128 bits déchiffrée.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsClientChalResp définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId et le message de **Réponse** transmet le résultat du paramètre response.

9.5.2.2.8 Message getAsSlotRk

C→H SymKey **getAsSlotRk**()

- Ce message lit la clé aléatoire pour la session de **Créneau AS** du **Client ECI**.

Description sémantique:

- Ce message est équivalent à la fonction de **sécurité évoluée** getAsSlotRk définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.9 Message getAsSessionRk

C→H SymKey **getAsSessionRk**(uint **sessionId,** uint **rkIndx)**

- Ce message lit la clé aléatoire actuelle (rkIndx == 0) et la clé aléatoire suivante (rkIndx == 1) pour la session du **Client ECI** portant l'identificateur sessionId.

Définitions des paramètres de la requête:

sessionId: uint	Identificateur de la session pour laquelle extraire la clé aléatoire de session.
rkIndx: uint	Détermine si la clé aléatoire de session actuelle (rkIndx == 0) ou suivante (rkIndx == 1) doit être extraite.

Description sémantique:

- Ce message est équivalent au message de sécurité évoluée getAsSessionRk défini dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.10 Message getAsSessionLimitCounter

C→H ulong **getAsSessionLimitCounter**(uint **sessionId**)

- Ce message renvoie la valeur du compteur de limite pour l'identificateur de session sessionId du **Client ECI**.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée getAsSessionLimitCounter définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

Définitions des paramètres de la requête:

sessionId: uint	Identificateur de la session pour laquelle extraire le compteur de limite.

9.5.2.2.11 Message setAsSessionLimitEvent

C→H ulong **setAsSessionLimitEvent** (uint **sessionId,** ulong **eventLimit**)

- Ce message définit la valeur limite eventLimit pour le champ limitCounter de la session du **Client ECI** portant l'identificateur sessionId pour qu'un message reqAsEventSessionLimit soit renvoyé au **Client ECI**.

Définitions des paramètres de la requête:

sessionId: uint	Identificateur de la session pour laquelle définir le paramètre eventLimit.
eventLimit: ulong	Valeur de la limite d'événement à définir.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée setAsSessionLimitEvent définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.12 Message reqAsEventSessionLimit

H→C reqAsEventSessionLimit (uint **sessionId**)
C→H resAsEventSessionLimit ()

- Ce message renvoie la valeur du compteur de limite pour l'identificateur de session sessionId du **Client ECI**.

Définitions des paramètres de la réponse:

sessionId: uint	Identificateur de la session ayant généré un événement eventLimit.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée reqAsEventSessionLimit définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** supprime le paramètre slotId.

9.5.2.2.13 Message getAsClientRnd

C→H SymKey **getAsClientRnd**()

- Ce message renvoie un nombre aléatoire de 128 bits.

Description sémantique:

- Cette fonction est équivalente au message de sécurité évoluée getAsClientRnd défini dans la Recommandation [UIT-T J.1014].

9.5.2.2.14 Message getAsSC

C→H uint **getAsSC**(uint **sessionId**)

- Ce message renvoie le statut du champ de commande d'embrouillage actuel du contenu d'une session.

Définitions des paramètres de la requête:

sessionId: uint	Identificateur de la session pour laquelle extraire le champ de commande d'embrouillage actuel.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée getAsSC définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.2.15 Message reqAsEventSC

H→C reqAsEventSC(uint **sessionId**; uint **scramblingControlField**)
C→H res**AsEventSC()**

- Ce message indique un changement du champ de commande d'embrouillage dans la session portant l'identificateur sessionId.

Définitions des paramètres de la réponse:

sessionId: uint	Identificateur de la session dans laquelle un changement du champ de statut d'embrouillage s'est produit.
scramblingControlField: uint	Nouvelle valeur du champ de statut d'embrouillage. Voir le § 9.9 de la Recommandation [UIT-T J.1014] pour la définition des valeurs et de leur sémantique.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsEventSC définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** supprime la valeur du paramètre slotId.

9.5.2.2.16 Message getChipsetId

C→H ulong **getChipsetId**()

- Ce message renvoie la valeur ChipsetID du **Bloc d'échelle de clés** comme défini dans la Recommandation [UIT-T J.1014].

9.5.2.2.17 Message getImageTargetId

C→H ECI_Image_Target_Id **getImageTargetId**()

- Ce message renvoie la valeur ECI_Image_Target_Id de l'équipement CPE comme défini dans le Tableau 6.2.2.2-1.

9.5.2.3 Définitions des messages de l'API AS de déchiffrement

9.5.2.3.1 Généralités

L'API AS de déchiffrement fournit les messages répertoriés dans le Tableau 9.5.2.3.1-1.

Deux sessions de déchiffrement peuvent être associées afin de permettre l'utilisation de différents mots de contrôle pour déchiffrer deux flux de contenu devant être traités comme un seul élément de contenu après le déchiffrement.

EXEMPLE: Une chaîne sportive peut être diffusée avec plusieurs canaux sonores, en ne mettant à disposition un canal propre à une langue particulière que si un abonnement distinct est disponible pour le déchiffrer. Il n'est possible d'associer que deux sessions.

Tableau 9.5.2.3.1-1 – Messages de déchiffrement à sécurité évoluée

Message	Type	Sens	Étiquette	Description
reqAsAStartDecryptSession	A	H→C	0x0	Démarrer une session de déchiffrement dans le **Créneau AS** du **Client ECI**.
reqAsComputeDecrCw	A	H→C	0x1	Calculer un mot de contrôle de déchiffrement.
reqAsAuthDecrSlotConfig	A	H→C	0x2	Authentifier la configuration du créneau au moyen de mécanismes d'authentification (mode de déchiffrement).

9.5.2.3.2 Message reqAsStartDecryptSession

C→H reqAsAStartDecryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config,**
ScrambleMode **sm**) →
H→C resAsAStartDecryptSession(uint **sessionId)**

- Ce message démarre une session de déchiffrement dans le **Créneau AS** du **Client ECI**.

Définitions des paramètres de la requête:

mh: ushort	**Pointeur de média** pour lequel le contenu est déchiffré (utilisé par l'**Hôte ECI** pour associer le contenu à déchiffrer à la ressource de déchiffrement attribuée à la session).
spk: PubKey	Clé publique de l'émetteur pour cette session.
config: SessionConfig	Configuration de la session.
sm: ScrambleMode	Mode de désembrouillage à utiliser. Voir le Tableau 9.5.2.3.2-1 pour la définition. Voir la NOTE.
NOTE – L'information incluse dans le paramètre sm ne doit pas contredire le paramètre cwUri d'un message reqAsComputeDecrCw ultérieur.	

Tableau 9.5.2.3.2-1 – Définition de ScrambleMode

```
typedef ScrambleMode {
 uchar      modeRef;
 uchar      mode[16] ;
} ScrambleMode;
```

La définition de **modeRef** est fournie dans le Tableau 9.9.2.11-1.

Tableau 9.5.2.3.2-2 – Définition de modeRef

Nom	Valeur	Description
ScrambleModeHost	0x01	L'hôte sélectionnera le mode d'embrouillage (ou de désembrouillage) en fonction d'informations normalisées ou propriétaires.
ScrambleModeDvb	0x02	La définition DVB pour le mode d'embrouillage est utilisée. L'octet 0 du champ de mode contient une valeur ayant la même signification que dans le champ scrambling_mode du descripteur Scrambling_descriptor défini dans la norme [IEC 62766-5-2]. L'octet 1 a la signification suivante pour les valeurs 0x02, 0x03 ou 0x10 de l'octet 0 (soit les modes DVB CSA1/2 et DVB CSA3 pour le désembrouillage et le mode DVB-CISSA version 1): Valeur == 0x01: (dés)embrouillage en mode TS. Valeur == 0x02: (dés)embrouillage en mode PES. Toutes les autres valeurs sont réservées; tous les octets non utilisés du champ de mode sont réservés. Voir la NOTE 1.
ScrambleModeCencEnum	0x03	Le mode d'embrouillage est défini dans la Recommandation [UIT-T T.871] ou l'octet 0 du champ de mode est défini comme suit: Valeur == 0x01: mode CENC CTR. Valeur == 0x02: mode CENC CBC. Les autres valeurs de l'octet 0 sont réservées. Pour les valeurs de l'octet 0 définies ci-dessus, l'octet 1 indique le sous-système: Valeur == 0x01: défini par l'hôte, pour un chiffrement sélectionné parmi les valeurs définies ci-dessous. Value == 0x02: chiffrement de segment complet comme défini dans le document [W3C GIF V89a]. Value == 0x03: chiffrement de sous-échantillon comme défini dans le document [W3C PNG]. Les autres valeurs pour l'octet 1 sont réservées. Pour les autres valeurs de l'octet 0, l'octet 1 est réservé. Les octets 2 à 15 sont réservés. Voir la NOTE 2.
RFU	Autre	Réservé à une utilisation future.
NOTE 1 – L'**Hôte ECI** prendra au moins en charge les modes DVB CSA1/2 et DVB CSA3 pour le désembrouillage et le mode DVB CISSA version 1 pour l'embrouillage et le désembrouillage. NOTE 2 – Le **Client ECI** ou (s'il y est autorisé) l'**Hôte ECI** peut sélectionner un mode d'embrouillage pour le chiffrement qui soit adapté à l'application, en prenant notamment en compte le fait que les applications de type streaming utilisent généralement le chiffrement de segment complet CBC et les applications de stockage le mode CTR en bénéficiant éventuellement du chiffrement de sous-échantillon.		

Définitions des paramètres de la réponse:

sessionId: uint	Identificateur de la session créée.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsAStartDecryptSession définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId et le résultat de l'ID de session (sessionId) est renvoyé dans le message de **Réponse**.

L'**Hôte ECI** émettra également une fonction reqAsCoupleDecryptSession (Recommandation [UIT-T J.1014]) lorsqu'une deuxième session de déchiffrement de **Créneau AS** est lancée pour le même **Pointeur de média**, en vue d'associer la seconde à la première.

9.5.2.3.3 Message reqAsComputeDecrCw

C→H reqAsComputeDecrCw(int **sessionId**, ulong **cwUri,** uint **nSpk**, uint **nElk**, SymKey **elk**[24], PubKey **spk**[16], PubKey **popk**[16], SessionConfig **config**[16], uchar **XT**[32], uint **rkIndx,** Field2 **field2,** uint **cwIndx**) →
H→C resAsComputeDecrCw ()

- Ce message calcule un mot de contrôle de déchiffrement.

Définitions des paramètres de la requête:

sessionId: int	Identificateur de la session pour laquelle calculer un mot de contrôle.
cwUri: ulong	cwUri définit les applications du mot de contrôle. Les valeurs de cwUri sont définies au § 7.5 de la Recommandation [UIT-T J.1014].
nSpk: uint	Nombre de valeurs de Clé publique de l'émetteur dans le vecteur de Clés publiques de l'émetteur.
nElk: uint	Nombre de valeurs Elk dans le vecteur d'ELK.
elk[24]: SymKey	Vecteur des valeurs de clés chiffrées de manière symétrique devant être déchiffrées successivement par le mécanisme d'échelle de clés. La valeur elk[nElk-2] est l'entrée field1 de l'authentification des propriétés de contenu telle que définie au § 8.2.3 de la Recommandation [UIT-T J.1014] en utilisant la fonction définie au § 8.2.4.7 de la Recommandation [UIT-T J.1014].
spk[16]: PubKey	Vecteur de Clés publiques de l'émetteur tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
popk[16]: PubKey	Vecteur de Clés publiques d'opérateurs de plate-forme tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
config[16]: SessionConfig	Vecteur de configurations de session de client tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
XT[32]: uchar	Entrée supplémentaire du mécanisme de mot de contrôle tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
rkIndx: uint	Détermine si la clé aléatoire de session actuelle (rkIndx == 0) ou suivante (rkIndx == 1) doit être utilisée dans le calcul du mot de contrôle.
field2: Field2	Propriétés de contenu étendues non authentifiées dans le champ field1, comme défini au § 8.2.3 de la Recommandation [UIT-T J.1014].
cwIndx: uint	Index du mot de contrôle à calculer: 0 pour un mot de contrôle pair et 1 pour un mot de contrôle impair; pas de signification en cas de déchiffrement basé sur fichier.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsComputeDecrCw définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.3.4 Message reqAsAuthDecrSlotConfig

C→H reqAsAuthDecrSlotConfig(uint **sessionId**, InputV **inputV**; uchar **nSpk,** uint **spkIndx**, PubKey **spk**[16], PubKey **popk**[16], SessionConfig **cnf**[16], ulong **spkUri**, uchar **XT**[32], bool **online,** uchar **verifier[16]**) →
H→C resAsAuthDecrSlotConfig ()

- Ce message authentifie la configuration du créneau au moyen de mécanismes d'authentification (mode de déchiffrement).

Définitions des paramètres de la requête:

sessionId: uint	ID de la session pour laquelle authentifier la configuration de créneau.
inputV: InputV	Message contenant la valeur r chiffrée avec la clé publique du jeu de puces et protégée par la signature à clé secrète de l'émetteur, laquelle valeur est utilisée pour calculer la clé d'authentification servant à authentifier la configuration du **Créneau AS**.
nSpk: uchar	Nombre de valeurs de Clé publique de l'émetteur dans le vecteur de Clés publiques de l'émetteur.
spkIndx: uint	Index définissant l'emplacement de la Clé publique d'émetteur du Créneau AS dans le vecteur de Clés publiques de l'émetteur, la valeur de la Clé publique d'opération de plate-forme du Créneau AS dans le vecteur de Clés publiques d'opération de plate-forme et le champ slotConfig du vecteur clCnf utilisé par la suite pour calculer la clé d'authentification du client, voir la Recommandation [UIT-T J.1014].
spk[16]: PubKey	Vecteur de Clés publiques de l'émetteur tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
popk[16]: PubKey	Vecteur de Clés publiques d'opérateurs de plate-forme tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
cnf[16]: SessionConfig	Vecteur de configurations de client tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
spkUri: ulong	Règles d'utilisation du vecteur de Clés publiques de l'émetteur, utilisé par la suite pour calculer la clé d'authentification AK, voir la Recommandation [UIT-T J.1014].
XT[32]: uchar	Valeur du champ d'extension, utilisée pour calculer la clé d'authentification du client; voir la Recommandation [UIT-T J.1014]. La valeur par défaut est { 0x00 }.
online: bool	Si la valeur est "true", la clé aléatoire du créneau est utilisée pour le calcul de la clé d'authentification, obligeant le fournisseur à procéder à un nouveau calcul de clé d'authentification.
verifier[16]: uchar	Valeur avec laquelle reqAsAuthDecrSlotConfig authentifie la configuration de créneau.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsAuthDecrSlotConfig définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.4 API AS d'exportation

9.5.2.4.1 Généralités

L'API AS d'exportation fournit les messages répertoriés dans le Tableau 9.5.2.4.1-1.

Tableau 9.5.2.4.1-1 – Messages d'exportation à sécurité évoluée

Message	Type	Sens	Étiquette	Description
reqAsExportConnSetup	A	C→H	0x0	Configurer une **Connexion d'exportation** entre des sessions de déchiffrement et de chiffrement.
reqAsExportConnEnd	A	C→H	0x1	Terminer la session d'exportation existante.

9.5.2.4.2 Message reqAsExportConnSetup

C→H reqAsExportConnSetup(uint **sessId,** ushort **expMh,** uint **grpIndx**; CertSerialChain **expCh,** CertSerialChain **impCh,** CertSerialChain **auth**[]) →
H→C resAsExportConnSetup()

- Ce message configure une connexion de type sécurité évoluée entre la session de déchiffrement et la session de **Pointeur de média** d'exportation.

Définitions des paramètres de la requête:

sessId: uint	ID de la session d'exportation du **Créneau AS** du **Client ECI**.
expMh: ushort	Identificateur du **Pointeur de média** d'exportation devant être utilisé pour le chiffrement du contenu déchiffré dans les sessions AS.
grpIndx: uint	Index stockant la connexion de la session d'exportation; les valeurs autorisées sont 0 et 1. Ce paramètre peut être utilisé pour transférer l'authentification de la **Connexion d'exportation** à un **Micro serveur** (par exemple en prévision du changement à venir de l'identificateur de **Groupe d'exportation** dans un flux).
expCh: CertSerialChain	**Chaîne d'exportation** pour le **Client ECI**.
impCh: CertSerialChain	**Chaîne d'importation** pour le chiffrement/l'importation du **Client ECI**.
auth[]: CertSerialChain	**Certificats** d'autorisation pour la **Chaîne d'importation**.

Description sémantique:

- Ce message est équivalent à la fonction de **Sécurité évoluée** reqAsExportConnSetup définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur des paramètres slotId, impSlotId et ImpSessId. L'**Hôte ECI** utilisera le **Pointeur de média** de la session d'exportation pour connecter la session de déchiffrement AS à la session de chiffrement AS correspondante; en d'autres termes, il fournira les paramètres impSlotId et impSessId dans la fonction de sécurité évoluée reqAsExportConnSetup définie dans la Recommandation [UIT-T J.1014].

9.5.2.4.3 Message reqAsExportConnEnd

C→H reqAsExportConnEnd(ushort **expMh) →**
H→C resAsExportConnEnd()

- Ce message termine une session d'exportation existante.

Définitions des paramètres de la requête:

expMh: ushort	Session de **Pointeur de média** d'exportation des sessions AS pour laquelle l'échange de contenu se terminera.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsExportConnEnd définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur des paramètres slotId et sessionId associés à expMh.

9.5.2.5 API AS de chiffrement

9.5.2.5.1 Généralités

L'API AS de chiffrement fournit les messages répertoriés dans le Tableau 9.5.2.5.1-1.

Tableau 9.5.2.5.1-1 – Messages de chiffrement à sécurité évoluée

Message	Type	Sens	Étiquette	Description
reqAsStartEncryptSession	A	C→H	0x0	Démarrer une session de chiffrement.
reqAsComputeEncrCw	A	C→H	0x1	Calculer un mot de contrôle de chiffrement.
reqAsAuthEncrSlotConfig	A	C→H	0x2	Authentifier la configuration du créneau et les paramètres de chiffrement au moyen de mécanismes d'authentification (mode de chiffrement).
reqAsLdUssk	A	C→H	0x3	Charger la clé secrète du **Micro serveur**.
reqAsMInikLk1	A	C→H	0x4	Calculer le message d'initialisation asymétrique du **Micro client**.
reqAsEventCpChange	A	H→C	0x5	Message d'événement signalant un changement dans les propriétés du contenu importé dans une session de chiffrement.
setAsPermitCPChange	S	C→H	0x6	Activer/désactiver le changement des propriétés du contenu importé influant sur la sélection des mots de contrôle en vue du chiffrement dans une session de chiffrement.
setAsSC	S	C→H	0x7	Définir le champ de commande d'embrouillage du contenu chiffré d'une session de chiffrement.

9.5.2.5.2 Définition de chaînes cibles de client

Les **Micro serveurs** peuvent utiliser le **Système de traitement des Certificats** pour assurer une implémentation solide de l'authentification asymétrique du client. L'**ECI** définit des chaînes de certificats pour autoriser ce type d'authentification des **Micro clients**. Ces chaînes cibles sont utilisées comme entrées du message reqAsMInikLk1.

Les **Chaînes de Certificats** seront conformes au § 5.4.1. Deux types de **Certificats** sont concernés:

- Un **Certificat de Micro client** authentifie un **Micro client** unique; la clé publique du **Certificat** sera identique à la clé publique du jeu de puces de l'**Équipement CPE du Micro client** si ce dernier est un **Client ECI**.
- Un **Certificat** de groupe cible authentifie un ou plusieurs groupes cibles ou des **Certificats de Micro client**.

Les opérateurs de **Système Micro DRM** peuvent utiliser le mécanisme de **Listes de révocation ECI** en vue de gérer de manière sécurisée l'évolution des **Micro clients** authentifiés pour un serveur.

NOTE – La tenue à jour des **Listes de révocation** relève de l'opérateur de **Système Micro DRM**.

L'identificateur du **Certificat** du groupe cible est défini dans le Tableau 9.5.2.5.2-1.

Tableau 9.5.2.5.2-1 – Définition de l'identificateur de Groupe cible

Syntaxe	Nbre de bits	Mnémonique
`ECI_Target_Group_Id {`		
`padding(4)`		
`type`	4	uimsbf
`target_group_id`	20	uimsbf
`target_group_version`	8	uimsbf
`}`		

Sémantique:

type: nombre entier	Valeur conforme au Tableau 5.1.3-1.
target_group_id: nombre entier	Numéro du **Groupe cible**, unique dans le contexte du **Père**.
target_group_version: nombre entier	Augmente au cas où le micro groupe modifie son **Certificat**.

L'identificateur du **Certificat** du **Micro client** est défini dans le Tableau 9.5.2.5.2-2.

Tableau 9.5.2.5.2-2 – Définition de l'identificateur de Micro client

Syntaxe	Nbre de bits	Mnémonique
`ECI_Micro_Client_Id {`		
`padding(4)`		
`type`	4	uimsbf
`micro_client_id`	20	uimsbf
`micro_client_version`	8	uimsbf
`}`		

Sémantique:

type: nombre entier	Valeur conforme au Tableau 5.1.3-1.
micro_client_id: nombre entier	Numéro du **Micro client**, unique dans le contexte du **Père**.
micro_client_version: nombre entier	Augmente au cas où le micro groupe modifie son **Certificat**.

9.5.2.5.3 Message reqAsStartEncryptSession

C→H reqAsStartEncryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config**, uint **nEncr**, PubKey **encrSpk[MaxSpkEncr]**, PubKey **encrPopk[MaxSpkEncr]**, ulong **encrCwUri**)→
H→C resAsStartEncryptSession()

- Ce message démarre la session de chiffrement.

Définitions des paramètres de la requête:

mh: ushort	Identificateur du **Pointeur de média** du contenu chiffré pour lequel créer une session de chiffrement.
spk: PubKey	Clé publique de l'émetteur utilisée par le système de sécurité évoluée pour authentifier un message chiffré par l'émetteur et la clé LK1.
config: SessionConfig	Configuration de la session.
nEncr: uint	Nombre de valeurs de Clé publique de l'émetteur (et de Clé publique d'opération de plate-forme) supplémentaires définies pour un chiffrement et un éventuel déchiffrement ultérieur. La valeur maximale est MaxEncr (voir la Recommandation [UIT-T J.1014]).
encrSpk: PubKey[]	Vecteur comportant des valeurs de Clé publique de l'émetteur supplémentaires pour chiffrement.
encrPopk: PubKey[]	Vecteur comportant des valeurs de Clé publique d'opération de plate-forme supplémentaires pour chiffrement.
encrCwUri: ulong	Valeur CwUri à utiliser pour le chiffrement; voir le § 8.2.2 de la Recommandation [UIT-T J.1014].

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsStartEncryptSession définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId. L'**Hôte ECI** obtiendra les paramètres importSlotId et importSessionId à partir de la valeur mh.

NOTE – Le message de **Réponse** renvoie le nouvel identificateur de session créé si aucune erreur ne s'est produite.

9.5.2.5.4 Message reqAsComputeEncrCw

C→H reqAsComputeEncrCw(int **sessId**, ulong **cwUri**, uint **nElk**, SymKey **elk**[24], uchar **XT**[32],
uint **rkIndx.** Field2 **field2.** uint **cwIndx)**→
H→C resAsComputeEncrCw()

- Ce message calcule un mot de contrôle de chiffrement.

Définitions des paramètres de la requête:

sessId: int	Identificateur de la session pour calculer un mot de contrôle.
cwUri: ulong	cwUri définit les applications du mot de contrôle. Les valeurs de cwUri sont définies au § 7.5 de la Recommandation [UIT-T J.1014].
nElk: uint	Nombre de valeurs Elk dans le vecteur d'ELK.
elk[24]: SymKey	Vecteur des valeurs de clés chiffrées de manière symétrique devant être déchiffrées successivement par le mécanisme d'échelle de clés. La valeur elk[nElk-2] est l'entrée field1 de l'authentification des propriétés de contenu telle que définie au § 8.2.3 de la Recommandation [UIT-T J.1014] en utilisant la fonction définie au § 8.2.4.6 de la Recommandation [UIT-T J.1014].
XT[32]: uchar	Entrée supplémentaire du mécanisme de mot de contrôle tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
rkIndx: uint	Détermine si la clé aléatoire de session actuelle (rkIndx == 0) ou suivante (rkIndx == 1) doit être utilisée dans le calcul du mot de contrôle.
field2: Field2	Propriétés de contenu étendues non authentifiées dans le champ field1, comme défini au § 8.2.3 de la Recommandation [UIT-T J.1014].
cwIndx: uint	Index du mot de contrôle à calculer: 0 pour un mot de contrôle pair et 1 pour un mot de contrôle impair; pas de signification pour le chiffrement basé sur fichier.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsComputeEncrCw définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.5 Message reqAsAuthEncrSlotConfig

C→H reqAsAuthEncrSlotConfig(uint **sessId**, InputV **inputV**, uchar **XT**[32], bool **online,** uchar **verifier[16]**) →
H→C resAsAuthEncrSlotConfig()

- Ce message authentifie la configuration du créneau au moyen de mécanismes d'authentification (mode de chiffrement).

Définitions des paramètres de la requête:

sessId: uint	Identificateur de la session pour laquelle la configuration sera authentifiée.
inputV: InputV	Message contenant la valeur r chiffrée avec la clé publique du jeu de puces et protégée par la signature à clé secrète de l'émetteur, laquelle valeur est utilisée pour calculer la clé d'authentification servant à authentifier la configuration du **Créneau AS**.
XT[32]: uchar	Entrée supplémentaire du mécanisme de mot de contrôle tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
online: bool	Si la valeur est "true", la clé aléatoire du créneau est utilisée pour le calcul de la clé d'authentification, obligeant le fournisseur à procéder à un nouveau calcul de clé d'authentification.
verifier[16]: uchar	reqAsAuthDecrSlotConfig utilise cette valeur pour authentifier la configuration de créneau.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsAuthEncrConfig définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.6 Message reqAsLdUssk

C→H reqAsLdUssk(uint **sessId**, InputV **inputV,** uchar **XT**[32], bool **online,** uchar **mUssk**[NUSSK])→
H→C resAsLdUssk()

- Ce message charge la clé secrète du **Micro serveur** dans le cas d'une authentification asymétrique de **Clients ECI** qui seront capables de décoder le contenu.

Définitions des paramètres de la requête:

sessId: uint	Identificateur de la session pour laquelle la clé secrète du **Micro serveur** sera chargée.
inputV: InputV	Message contenant la valeur r chiffrée avec la clé publique du jeu de puces et protégée par la signature à clé secrète de l'émetteur, laquelle valeur est utilisée pour calculer la clé d'authentification servant à déchiffrer la clé secrète du **Micro serveur** devant être chargée.
XT[32]: uchar	Entrée supplémentaire du mécanisme de mot de contrôle tel que défini au § 7.5 de la Recommandation [UIT-T J.1014].
online: bool	Si la valeur est "true", la clé aléatoire du créneau est utilisée pour le calcul de la clé d'authentification, obligeant le fournisseur à procéder à un nouveau calcul de clé d'authentification.
mUssk[NUSSK]: uchar	Clé secrète chiffrée du **Micro serveur**.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée reqAsLdUssk définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.7 Message reqAsMInikLk1

C→H reqAsMInikLk1(uint **sessId**, ECI_certificate_Chain **ClCPK**) →
H→C resAsMInikLk1(InputV **inputV)**

- Ce message calcule le message d'initialisation asymétrique du **Micro client**.

Définitions des paramètres de la requête:

sessId: uint	Identificateur de la session pour laquelle la clé secrète du **Micro serveur** sera chargée.
ClCPK: ECI_certificate_Chain	**Chaîne de certificats** cible telle que définie dans le § 9.5.2.5.2 destinée à charger la clé publique du jeu de puces du **Micro client** devant être utilisée pour chiffrer la clé secrète de session entre le **Micro client** et le **Micro serveur**.

Définitions des paramètres de la réponse:

inputV: InputV	Clé de session Micro DRM chiffrée avec la clé publique du jeu de puces du **Micro client** et signée par la clé secrète du **Micro serveur**. Peut être utilisé par le Micro client comme message permettant de charger la clé de session commune LK_1.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée reqAsMInikLk1 définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.8 Message reqAsEventCpChange

H→C reqAsEventCpChange(int **sessionId**)

- Ce message demande un changement dans les propriétés du contenu importé dans une session de chiffrement.

Définitions des paramètres de la requête:

sessionId: int	Session de chiffrement lors de laquelle s'est produit un événement de changement des propriétés du contenu importé.

Description sémantique:

- Ce message est équivalent à la fonction de sécurité évoluée reqAsEventSC définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** supprime le paramètre slotId.

9.5.2.5.9 Message setAsPermitCPChange

C→H setAsPermitCPChange(int **sessionId**; bool **permit**)

- Ce message initie un changement des propriétés du contenu importé dans une session de chiffrement.

Définitions des paramètres de la requête:

sessionId: int	Session de chiffrement pour autoriser un transfert automatique du mot de contrôle lors d'un changement des propriétés du contenu prévu ou en attente.
permit: bool	La valeur "true" signifie que l'autorisation est accordée, la valeur "false" qu'elle est refusée.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée setAsPermitCPChange définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.10 Message setAsSC

C→H setAsSC(int **sessionId**, uint **scramblingControlField**)

- Ce message définit la prochaine valeur du champ de commande d'embrouillage dans la session de chiffrement.

Définitions des paramètres de la requête:

sessionId: int	Session de chiffrement pour laquelle définir le champ de commande d'embrouillage devant être utilisé à la première possibilité de changement dans le flux.
scramblingControlField: uint	Valeur du champ de commande d'embrouillage; voir le § 9.9 de la Recommandation [UIT-T J.1014] pour les valeurs autorisées et leur signification.

Description sémantique:

- Cette fonction est équivalente à la fonction de sécurité évoluée setAsSC définie dans la Recommandation [UIT-T J.1014]; l'**Hôte ECI** fournit la valeur du paramètre slotId.

9.5.2.5.11 Codes d'erreur de l'API de sécurité évoluée (AS)

Tous les codes d'erreur de l'API AS sont définis au § 8.2.4.15 de la Recommandation [UIT- J.1014].

9.5.3 API de Carte à puce

9.5.3.1 Introduction

L'**ECI** permet aux **Clients ECI** de créer une interface avec un module de sécurité local détachable unique (**Carte à puce**). Les **Clients ECI** peuvent établir un canal sécurisé du **Client ECI** à la **Carte à puce** ou directement (sur le plan de la sécurité) de la **Carte à puce** au bloc de sécurité évoluée, afin d'assurer une protection maximale des mots de contrôle. Les détails des protocoles d'échange utilisés dans la gestion des clés ne sont pas définis par l'**ECI** mais entièrement par le système CA/DRM, sur la base de l'API du bloc de **sécurité évoluée**, comme défini dans la Recommandation [UIT-T J.1014].

Les **Équipements CPE** conformes **ECI** peuvent posséder un ou plusieurs emplacements de lecteur de carte. L'**Hôte ECI** gère les lecteurs de carte de manière complètement transparente pour les **Clients ECI**. Il associe les **Cartes à puce** insérées aux **Clients ECI** disponibles. A cet effet, les **Clients ECI** publient une liste de spécificateurs de carte à destination de l'**Hôte ECI**. Ce dernier gère tout conflit potentiel entre les **Clients ECI** souhaitant se connecter à la même **Carte à puce**. L'**Hôte ECI** assure par ailleurs la gestion des contentions pour les lecteurs de carte.

9.5.3.2 Spécifications de base

Le présent paragraphe expose les normes et spécifications de base auxquelles le matériel du lecteur de carte de l'**Équipement CPE** ainsi que les pilotes et les logiciels de l'**Hôte ECI** associés doivent se conformer.

Les caractéristiques physiques du lecteur de carte d'un **Équipement CPE** peuvent être basées sur les exigences du marché en vigueur. Le format le plus répandu pour les cartes d'accès conditionnel est ID-1 (taille d'une carte de crédit), mais les cartes au format ID-000 (SIM) sont également utilisées. Voir les normes [ISO/CEI 7816-1], [ISO/CEI 7816-2] et [ISO/CEI 14496-12] pour référence.

Le lecteur de carte d'un **Équipement CPE** doit être conforme au § 5 de la norme [ISO/CEI 7816-3] et prendre au moins en charge les conditions de fonctionnement de classe A (5V) et B (3V). Les broches suivantes doivent être prises en charge: C1 (VCC), C2 (RST), C3 (CLK), C5 (GND) et C7 (I/O).

Les **Hôtes ECI** peuvent prendre en charge des lecteurs de carte ne répondant pas à ces normes. Ces types de lecteurs doivent être clairement signalés et ne doivent pas être confondus par l'**Utilisateur** avec un lecteur de carte **ECI** ordinaire.

Le matériel de l'**Hôte ECI** et du lecteur de carte de l'**Équipement CPE** doit prendre en charge les fonctionnalités propres à l'**ECI** définies aux § 6 à 12 de la norme [ISO/CEI 7816-2]. L'**Hôte ECI** initialisera toute carte insérée au moyen des procédures définies dans la norme [ISO/CEI 7816-2].

L'**Hôte ECI** implémentera les fonctionnalités énoncées dans la norme [ISO/CEI 7816-3] selon les besoins de l'application des spécifications de la présente Recommandation. L'**Hôte ECI** prendra en charge la norme [ISO/CEI 7816-5] autant que nécessaire pour prendre en charge la fonctionnalité d'extraction de l'AID définie au § 9.5.3.3 ci-après.

9.5.3.3 Gestion de l'accès à la Carte à puce

Avant d'initialiser une connexion vers un **Client ECI**, l'**Hôte ECI** initialisera le protocole et le lecteur de carte conformément aux § 6 à 11 de la norme [ISO/CEI 7816-3]. Il sélectionnera les configurations appropriées pour le protocole, les paramètres de synchronisation des communications et la classe de fonctionnement de la **Carte à puce**.

L'**Hôte ECI** doit pouvoir extraire l'identificateur d'application (AID) (tel que défini au § 8.2.1.2 de la norme [ISO/CEI 7816-4]), conformément au § 8.2.1 de la norme [ISO/CEI 7816-4], à partir de la carte, à savoir dans les octets historiques, comme défini au § 8.2.2.1 de la norme [ISO/CEI 7816-4], ou dans la chaîne de données initiale. En ce qui concerne les **Cartes à puce** multi-applications, l'**Hôte ECI** sera capable d'extraire la liste d'AID comme défini au § 8.2 de la norme [ISO/CEI 7816-4], en particulier aux § 8.2.1.1, 8.2.2 et 8.2.2.3.

L'**Hôte ECI** utilisera la liste suivante d'identificateurs de carte:

1) S'il s'agit d'une carte multi-applications conformément à la norme [ISO/CEI 7816-4], elle utilisera comme liste d'identificateurs de carte la liste d'AID extraite des modèles d'application du fichier EF.DIR ainsi que les AID directement représentés dans le fichier EF.DIR.
2) S'il ne s'agit pas d'une carte multi-applications conformément à l'alinéa 1) ci-dessus, l'AID extrait des "octets historiques" comme défini au § 8.1.1 ou 8.1.2 de la norme [ISO/CEI 7816-4] sera utilisé comme identificateur unique de la carte.
3) Si aucun AID ne peut être extrait conformément aux alinéas 1) et 2) ci-dessus, l'ATR (Answer-to-Reset, réponse à la remise à zéro) définie au § 8.2 de la norme [ISO/CEI 7816-4] sera utilisée comme identificateur unique de la carte. A des fins de mise en correspondance, l'ATR est définie de T0 à Tk, à l'exception de l'octet TCK (s'il est présent).

Sur la base de la liste d'identificateurs de carte ci-dessus, l'**Équipement CPE** effectuera la mise en correspondance avec les **Clients ECI** en conséquence.

Un **Client ECI** fournira la liste des spécificateurs d'identificateur de carte éligibles s'il est prêt à se connecter à une carte. L'attribut d'exclusivité de la carte sera présent pour chaque spécificateur d'identificateur de carte et indique que l'**Hôte ECI** signalera à l'**Utilisateur** un conflit de résolution d'accès à la **Carte à puce**. Ceci s'applique si plusieurs **Clients ECI** demandent à accéder à une **Carte à puce** correspondant au spécificateur d'identificateur de carte et insérée ou présente dans un des lecteurs de **Carte à puce** de l'**Équipement CPE**.

L'**Hôte ECI** détectera et, dans la mesure du possible, résoudra tout conflit entre un identificateur de carte et les **Clients ECI** correspondants conformément aux règles suivantes:

- Une **Carte à puce** est considérée comme correspondant à un **Client ECI** si l'un des identificateurs de carte de sa liste d'identificateurs correspond à l'un des spécificateurs d'identificateur de carte du **Client ECI**.
- Si une **Carte à puce** correspond à plusieurs **Clients ECI** et qu'aucun de ces clients ne souhaite un accès exclusif, une session de carte est accordée dans l'ordre suivant:
  – Une session de carte sera établie en premier lieu pour le **Client ECI** avec lequel une session a été établie le plus récemment.
  – Si aucun des **Clients ECI** ne se trouve dans ce cas ou si la carte n'est pas reconnue comme ayant été insérée dans le lecteur de carte de l'**Équipement CPE** précédemment, une session de carte sera établie selon un algorithme choisi par l'**Hôte ECI**.
- Un **Client ECI** n'étant pas en mesure d'utiliser la **Carte à puce** déconnectera la session de **Carte à puce** afin que l'**Hôte ECI** puisse la mettre en correspondance avec d'autres **Clients ECI**, qui pourront tenter de s'y connecter.

Les **Clients ECI** doivent être capables de traiter des événements de type "connexion" et "déconnexion" générés par l'**Hôte ECI** lors d'une session de **Carte à puce**.

9.5.3.4 Gestion des contentions du lecteur de Carte à puce

Ce paragraphe définit les fonctionnalités des **Hôtes ECI** liées à la résolution des conflits d'applications et destinées à gérer les contentions entre les clients et les lecteurs de carte disponibles quant à l'accès aux **Cartes à puce**.

Lorsqu'un **Client ECI** accèdera aux **Cartes à puce** par l'intermédiaire d'un lecteur de carte (session de **Carte à puce**), il leur donnera la priorité de session. Les valeurs possibles sont les suivantes:

– **Active**: utilisée pour les fonctions principales qui occasionnent une gêne pour l'**Utilisateur** si elles sont interrompues. Par exemple, une session de visionnement demandée par l'**Utilisateur** ou une session d'enregistrement programmée au préalable par l'**Utilisateur**.

– **Arrière-plan**: utilisée pour les processus en arrière-plan pouvant être interrompus si nécessaire; il s'agit de la priorité par défaut. Par exemple, le traitement de messages EMM pour l'acquisition des futurs droits d'accès.

Un **Client ECI** sera capable de demander l'insertion d'une **Carte à puce** – impliquant une utilisation active – en faisant référence à un ou plusieurs **Pointeurs de média** ou à une chaîne indiquant quelle application demande la carte si cette dernière n'est pas sollicitée pour un **Pointeur de média** particulier.

L'**Hôte ECI** orientera l'**Utilisateur** vers un lecteur de carte approprié si un **Client ECI** demande l'accès à une carte, en suivant les lignes directrices suivantes:

– Orienter vers un lecteur de carte libre, s'il y en a un.

– Orienter vers un lecteur en mode arrière-plan si aucun lecteur de carte libre n'est disponible.

– Si aucun lecteur en mode arrière-plan ou lecteur libre n'est disponible, l'**Hôte ECI** s'efforcera d'orienter l'**Utilisateur** vers un lecteur en mode actif qui lui cause le moins de gêne en utilisant les informations issues de l'application/du **Client ECI** sur les sessions actuellement actives de ces lecteurs.

Le processus ci-dessus peut amener l'**Hôte ECI** à utiliser des informations supplémentaires pour mettre en correspondance la carte avec le bon type de lecteur (par exemple dimensions physiques), en associant un type de lecteur au **Client ECI** répondant aux exigences d'une connexion réussie – en supposant que le même type de carte sera inséré à l'avenir. L'**Hôte ECI** peut utiliser ses propres politiques à cet effet.

9.5.3.5 API de gestion des sessions de Carte à puce

9.5.3.5.1 Généralités

L'API de gestion des sessions de **Carte à puce** fournira aux clients un accès géré aux **Cartes à puce** définies aux § 9.5.3.3 et 9.5.3.4.

Les messages d'API disponibles pour la gestion des sessions de **Carte à puce** sont répertoriés dans le Tableau 9.5.3.5.1-1.

Tableau 9.5.3.5.1-1 – Messages de l'API de gestion des sessions de Carte à puce

Message	Type	Sens	Étiquette	Description
setCardMatch	set	C→H	0x0	Définir la liste de spécificateurs d'identificateur de carte pour le **Client ECI**.
callCardSessionPrio	call	C→H	0x1	Définir la priorité de la session de **Carte à puce**.
getCardConnStatus	get	H→C	0x2	Fournit le statut de connexion de la carte.
reqCardConOpen	A	H→C	0x3	Informe le **Client ECI** qu'une session de carte a été ouverte.
reqCardConClose	A	H→C	0x4	Informe le **Client ECI** qu'une session de carte a été fermée.
reqCardConClose	A	C→H	0x5	Informe l'**Hôte ECI** que le **Client ECI** souhaite terminer une session avec la carte connectée.

9.5.3.5.2 Message setCardMatch

C→H setCardMatch(uint **matchListLength,** CardSpecifier **matchList[])**

- Ce message permet au **Client ECI** d'indiquer les identificateurs de carte auxquels il souhaite se connecter.

Définition de la propriété CardMatch

matchListLength: uint	Longueur de la liste de correspondance (matchList) en ce qui concerne les spécificateurs.
matchList: CardSpecifier[].	Voir le Tableau 9.5.3.6.1-1. Messages de communication avec la **Carte à puce**. L'**Hôte ECI** utilisera cette liste pour mettre en correspondance les **Cartes à puce** connectées avec le **Client ECI** conformément au § 9.5.3.3. La définition du type est indiquée dans le Tableau 9.5.3.5.2-1 et les valeurs du champ specifierType dans le Tableau 9.5.3.5.2-2.

Tableau 9.5.3.5.2-1 – Définitions du type de spécificateur de Carte à puce

```
#define MaxAtr 32
#define MaxAid 16

typedef struct CardSpecifier {
    bool exclusiveFlag;
    uchar specifierType;
    union specifier {
        struct {
            uchar atrLen;
            byte atr[MaxAtr];
        } atrSpec;
        struct {
            uchar aidLen;
            byte aid[MaxAid];
        } aidSpec;
    }
} CardSpecifier;
```

Tableau 9.5.3.5.2-2 – Type de spécificateur de Carte à puce

Nom	Valeur	Description
CardSpecifierATR	0x01	Le spécificateur de carte est de type ATR. Une carte correspond au spécificateur si le champ **atrLen** est identique à la longueur de l'ATR de la carte et si les octets de l'ATR de la carte correspondent aux premiers octets **atrLen** du champ **atr**. L'ATR d'une carte est définie au § 9.5.3.5.3, T0...TCK.
CardSpecifierAID	0x02	Le spécificateur de carte est de type AID. Une carte correspond au spécificateur si le champ **aidLen** est identique à la longueur de l'AID de la carte et si les octets de l'AID de la carte correspondent aux premiers octets **aidLen** du champ **aid**. L'AID d'une carte est défini au § 9.5.3.3.
RFU	autre	Réservé à une utilisation future.

Préconditions:

1) Le **Client ECI** est prêt à répondre aux messages invCardConOpen et invCardConClose si **matchListLength** > 0.

Postconditions:

1) L'**Hôte ECI** associera toute carte insérée dans un lecteur de carte au **Client ECI** conformément au § 9.5.3.3. En cas de correspondance, il ouvrira une session de carte avec le **Client ECI** comme défini au § 9.5.3.5.5.

2) L'**Hôte ECI** n'abandonnera pas une session de carte en cours si la nouvelle liste matchList n'assure plus de correspondance avec la **Carte à puce** actuellement connectée. Le **Client ECI** utilisera le message reqCardConnClose à cet effet.

9.5.3.5.3 Message callCardSessionPrio

C→H callCardSessionPrio(uchar **priority,** uint **nrMh,** ushort **mH[],** char ***clientApplication)**

- Ce message met à jour la priorité de la session de carte et fournit à l'**Hôte ECI** la liste de **Pointeurs de média** mH et la raison interne pour laquelle le **Client ECI** demande une session de carte ou présente une session de carte active.

Définition des paramètres d'appel

priority: uchar	Priorité de la session de carte demandée par le **Client ECI**. Les valeurs sont définies dans le Tableau 9.5.3.5.3-1.
nrMh: uint	Nombre de **Pointeurs de média** dépendant de l'établissement d'une session active avec la carte.
mH: ushort	Liste des **Pointeurs de média** demandant l'établissement d'une session active avec une **Carte à puce**.
clientApplication: char *	Chaîne terminée par un caractère Null indiquant la raison pour laquelle le **Client ECI** demande une session active avec une **Carte à puce** sans lien avec une activité de **Pointeur de média**. Si la valeur de ce pointeur est Null, aucune demande de ce type n'a été formulée. Si la valeur n'est pas Null, la chaîne comportera une valeur pertinente pour l'**Utilisateur**. Le nombre de caractères pouvant être affiché est limité à 40.

Tableau 9.5.3.5.3-1 – Valeurs de priorité d'une session de Carte à puce

Nom	Valeur	Description
CardPriorityBackground	0x01	La priorité requise par le **Client ECI** pour la carte est de type arrière-plan et est définie au § 9.5.3.4.
CardPriorityActive	0x02	La priorité requise par le **Client ECI** pour la carte est de type "active" et est définie au § 9.5.3.4.
RFU	autre	Réservé à une utilisation future.

Postconditions:

1) L'**Hôte ECI** gèrera la session de carte conformément au § 9.5.3.4 en fonction du paramètre **priority** et utilisera les paramètres **mH** et **clientApplication** pour résoudre les conflits d'accès aux lecteurs de carte par l'intermédiaire de l'interface d'**Utilisateur** si nécessaire.

9.5.3.5.4 Message getCardConnStatus

C→H uchar**getCardConStatus()**

- Ce message renvoie le statut actuel de connexion à une session de **Carte à puce**.

Définition de la propriété: voir le Tableau 9.5.3.5.4-1.

Tableau 9.5.3.5.4-1 – Valeurs du statut de connexion de la carte

Nom	Valeur	Description
CardConNo	0x00	Aucune session n'est établie entre le **Client ECI** et la **Carte à puce**.
CardConYes	0x01	Une session est établie entre le **Client ECI** et la **Carte à puce**.
RFU	autre	Réservé à une utilisation future.

9.5.3.5.5 Message reqCCardConOpen

H→C reqCCardConOpen()→
C→H resCardConOpen()

- **Ce message permet à** l'**Hôte ECI** d'informer le **Client ECI** d'un nouvel événement d'établissement de connexion (session) avec une carte; le **Client ECI** répond en confirmant que l'événement est en cours de traitement.

Préconditions de la requête:

1) Une session de carte doit être établie avec le **Client ECI** conformément au § 9.5.3.3.

Postconditions de la réponse:

1) Le **Client ECI** gèrera la priorité de la session conformément aux conditions précisées au § 9.5.3.4.
2) Le **Client ECI** fermera la session s'il n'a pas besoin de la carte, comme défini au § 9.5.3.3.

9.5.3.5.6 Message reqCCardConClose

H→C reqCCardConClose () →
C→H resCardConClose ()

- Ce message permet à l'**Hôte ECI** d'informer le **Client ECI** que la session de carte a été fermée. Le **Client ECI** répond en confirmant que l'événement a été traité.

Préconditions de la requête:

1) La carte a été retirée du lecteur ou un dysfonctionnement majeur du sous-système du lecteur de carte a entraîné la perte de la connexion.

Postconditions de la réponse:

1) La **Réponse** du **Client ECI** confirme qu'il a traité l'événement et est prêt à accepter une nouvelle connexion à une carte conformément à la propriété CardMatch.

9.5.3.5.7 Message reqHCardConClose

C→H reqHCardConClose() →
H→C reqHCardConClose ()

- Ce message permet au **Client ECI** d'indiquer à l'**Hôte ECI** qu'il n'a plus besoin d'interagir avec la **Carte à puce** connectée.

Postconditions de la réponse:

1) L'**Hôte ECI** connecte la **Carte à puce** à un autre **Client ECI** correspondant tel que défini au § 9.5.3.3 et ne tentera pas de connecter cette carte au **Client ECI** (réamorçages et cycles d'alimentation en attente).

2) L'**Hôte ECI** attendra d'avoir reçu la **Réponse** avant de reconnecter éventuellement une autre **Carte à puce** correspondante au **Client ECI**.

9.5.3.6 Définitions des messages de l'API de communication avec la Carte à puce

9.5.3.6.1 Généralités

L'API de commande et de **Réponse** de la **Carte à puce** fournira les primitives de la session de communication entre un **Client ECI** et une **Carte à puce** dans le cadre d'une session de **Carte à puce** ouverte gérée par l'**Hôte ECI**. Le **Client ECI** peut réaliser des échanges commande/réponse [ISO/CEI 7816-3] avec l'**Hôte ECI** au niveau des APDU (voir la NOTE) comme défini au § 12 de la norme [ISO/CEI 7816-3]. Le **Client ECI** a accès à toutes les fonctions de gestion de la **Carte à puce**, il peut effectuer des opérations de remise à zéro et de réinitialisation avec des configurations de paramètres personnalisées si besoin, et extrait les configurations de communication. Les messages de l'API d'**ECI** sont définis dans le Tableau 9.5.3.6.1-1.

NOTE – Cela permet également des échanges au niveau des TPDU via le protocole T=0 par des échanges de commandes et de réponses courtes au niveau de l'interface avec les APDU.

Tableau 9.5.3.6.1-1 – Messages de l'API de communication avec la Carte à puce

Message	Type	Sens	Étiquette	Description
reqCardCmdRes	A	C→H	0x6	Envoyer une commande à la carte, recevoir une réponse de la carte.
reqCardReInit	A	C→H	0x7	Effectuer une remise à zéro (à chaud ou à froid) de la carte et réexécuter la séquence d'initialisation avec le dernier paramètre de préférence d'initialisation.
callCardSetProp	set	H→C	0x8	Définir le paramètre de communication avec la carte.
callCardGetProp	get	H→C	0x9	Obtenir la propriété/le paramètre de communication avec la carte.

9.5.3.6.2 Message reqCardCmdRes

C→H reqCardCmdRes(byte **nodeAddrByte,** uint **cmdApduLen,** byte **cmdApdu[]) →**
H→C resCardCmdRes(uint **resApduLen,** byte **resApdu[])**

- Comme défini au § 12 de la norme [ISO/CEI 7816-3], ce message envoie une commande APDU à la **Carte à puce** via l'**Hôte ECI** et reçoit une réponse APDU. Les codes d'erreur associés sont définis dans le Tableau 9.5.3.6.2-1.

Définition des paramètres de la requête:

nodeAddrByte: byte	Octet NAD (Node Address, noeud d'adresse) si le protocole T=1 est établi comme protocole de la **Carte à puce** tel que défini au § 11.3.2.1 de la norme [ISO/CEI 7810]. Ce paramètre est ignoré si le protocole de la **Carte à puce** est défini sur T=0.
cmdApduLen: uint	Longueur de la commande APDU en octets. Il convient de noter que le codage interne de la longueur du paramètre **cmdApdu** ne doit pas dépasser **cmdApduLen.**
cmdApdu: byte []	Commande APDU à envoyer à la carte. L'**Hôte ECI** ignore les octets excédentaires du champ cmdApdu.

Définition des paramètres de la réponse:

resApduLen: uint	Longueur de la **Réponse** APDU en octets.
resAdpu: byte []	**Réponse** APDU reçue de la carte.

Préconditions de la requête:

1) Une session de **Carte à puce** est ouverte avec le **Client ECI**.
2) Le message reqCardCmdRes précédent a entraîné une réponse resCardCmdRes ou la connexion a été (ré-)initialisée.

Tableau 9.5.3.6.2-1 – Codes d'erreur du message resCardCmdRes

Nom	Description
ErrCardConnOpenNot	Voir le Tableau 9.5.3.7-1.
ErrCardConnFail	

9.5.3.6.3 Message reqCardReInit

C→H reqCardReInit(uchar **resetMode)** →
H→C resCardReInit()

- Ce message demande à l'**Hôte ECI** de remettre la **Carte à puce** à zéro à l'aide du paramètre resetMode et la réinitialise avec les derniers paramètres de préférence de connexion de la carte. La **Réponse** est envoyée quand le processus est terminé (ou a échoué). Les codes d'erreur associés sont définis dans le Tableau 9.5.3.6.3-2.

Définition des paramètres de la requête:

resetMode: uchar	Voir le Tableau 9.5.3.6.3-1.

Tableau 9.5.3.6.3-1 – Valeurs du paramètre resetMode de la carte

Nom	Valeur	Description
CardResetCold	0x01	Une remise à zéro à froid sera réalisée et la carte sera réinitialisée comme si elle était mise sous tension pour la première fois (voir le § 6.2.3 de la norme [ISO/CEI 7816-1]).
CardResetWarm	0x02	Une remise à zéro à chaud sera réalisée, les paramètres de synchronisation des communications de la carte seront réinitialisés (voir le § 6.2.3 de la norme [ISO/CEI 7816-3]); la sélection du protocole et des paramètres définie au § 9 de la norme [ISO/CEI 7816-3] sera effectuée à nouveau, le cas échéant. Elle peut être utilisée notamment pour tenter de passer les paramètres de synchronisation d'interface à la valeur préférée d'un **Client ECI**.
RFU	autre	Réservé à une utilisation future.

Préconditions de la requête:

1) Une session de **Carte à puce** est ouverte avec le **Client ECI**.

Postconditions de la réponse:

1) La **Réponse** indique que le protocole et les paramètres de l'interface ont été établis avec succès.

Tableau 9.5.3.6.3-2 – Codes d'erreur du message resCardCmdRes

Nom	Description
ErrCardConnOpenNot	Voir le Tableau 9.5.3.7-1.
ErrCardConnFail	

9.5.3.6.4 Message callCardSetProp

C→H callCardSetProp (ushort **propTag,** uin**t valueLen,** byte ***propValue)**

- Ce message définit sur **propValue** la propriété accessible en écriture indiquée par le paramètre **propTag** dans l'interface de la **Carte à puce**.

Définition des paramètres de la requête:

propTag: ushort	L'étiquette de la propriété relative au protocole de communication de la carte devant être changée. Les valeurs sont définies dans le Tableau 9.5.3.6.5-2.
valueLen: uint	Longueur du champ paramValue en octets.
propValue: byte *	Pointeur vers la valeur de propriété à écrire dans le paramètre indiqué par propTag.

Tableau 9.5.3.6.4-1 – Codes d'erreur du message callCardSetProp

Nom	Description
ErrCardConnOpenNot	Voir le Tableau 9.5.3.7-1.

9.5.3.6.5 Message callCardGetProp

C→H callCardGetProp(ushort **propTag,** uint **valueLen,** byte ***propValue)**

- Ce message lit la propriété accessible indiquée par le paramètre **propTag** de l'interface de la **Carte à puce** dans **propValue**. Les codes d'erreur associés sont définis dans le Tableau 9.5.3.6.5-1.

Définition des paramètres de la requête:

propTag: ushort	L'étiquette de la propriété relative au protocole de communication de la carte devant être changée. Les valeurs sont définies dans le Tableau 9.5.3.6.5-2.
valueLen: uint	Longueur maximale du champ propValue en octets. Les octets excédentaires de la propriété ne sont pas copiés dans propValue.
propValue: byte *	Pointeur vers la valeur de propriété demandée.

Tableau 9.5.3.6.5-1 – Codes d'erreur du message callCardSetProp

Nom	Description
ErrCardConnOpenNot	Voir le Tableau 9.5.3.7-1.

Tableau 9.5.3.6.5-2 – Valeurs des étiquettes de l'API de la carte et sémantique pour les propriétés de protocole de la carte

Nom	Valeur de l'étiquette	Description
CardPropClass	0x0001	Un octet. Classe A = 0x01, classe B = 0x02, Classe C = 0x03. Les autres valeurs sont réservées à une utilisation future. Lecture seule.
CardPropAtrLen	0x0002	Un octet. Longueur en octets de l'ATR de la carte indiquée dans **CardPropAtr**. Lecture seule.
CardPropAtr	0x0003	Chaîne d'octets, 16 octets max. ATR de la carte lors d'une remise à zéro à froid. Lecture seule.
CardPropPpsExch	0x0004	Si différent de 0x00, la carte et l'interface ont réalisé un échange de PPS avec succès. Lecture seule.
CardPropPpsVal	0x0004	Un octet. Valeur du résultat de l'échange de PPS de la carte pour PPS1. Les autres valeurs ne sont pas prévues dans la présente Recommandation. Lecture seule.
CardPropTAEff	0x0005	Un octet. La valeur effective de l'octet TA appliquée pour la fréquence d'horloge sur l'interface. Lecture seule.
CardPropTCEff	0x0006	Un octet. La valeur effective de l'octet TC appliquée pour la fréquence d'horloge sur l'interface. Lecture seule.
CardPropProt	0x0007	Un octet. Indique le protocole sélectionné par le dispositif d'interface pour communiquer avec la carte. Les valeurs sont définies au § 8.2.3 de la norme [ISO/CEI 7816-3], champ "T". La valeur 0x00 correspond au protocole T=0, la valeur 0x01 correspond au protocole T=1. D'autres valeurs peuvent également être présentes (jusqu'à 0x0E). Lecture seule.
CardPropT1IFSC	0x0008	Un octet. La valeur de protocole actuelle de l'IFSC (Taille de champ d'information de carte) dans le protocole T=1 encodée comme défini au § 11.4.2 de la norme [ISO/CEI 7816-3]. Lecture seule.
CardPropT1IFSD	0x0009	Un octet. La valeur de protocole actuelle de l'IFSD (Taille de champ d'information de dispositif = lecteur de carte) dans le protocole T=1 encodée comme défini au § 11.4.2 de la norme [ISO/CEI 7816-3]. Lecture seule.
CardPropAidListLen	0x000A	Un octet: longueur de la liste d'AID de la carte extraite de la carte lors de l'initialisation. Lecture seule.
CardPropAidList	0x000B	*(byte[MaxAid]): liste d'AID extraite de la carte lors de l'initialisation. Lecture seule.
CardPropClassPref	0x0011	Trois octets. Séquence de valeurs de classe préférées. Une tentative d'établissement des valeurs de préférence dans l'ordre (sans compromettre la sécurité) sera réalisée. Les valeurs des trois octets figurent dans **CardPropClass**, la valeur 0x00 signifiant "pas d'autre préférence". Lecture et écriture.
CardPropImplClock	0x0012	Une valeur de **TA** d'un octet doit être appliquée au cas où le bit 5 de l'octet $\mathbf{TA}_2$ de l'ATR indique des valeurs implicites pour la fréquence d'horloge. Lecture et écriture.
CardPropPps1SegLen	0x0013	Un octet. La valeur représente un nombre binaire non signé. La valeur minimale est 0, la valeur maximale est 0x08. Représente le nombre de valeurs PPS1 à essayer lors d'un échange PPS de négociation dans la propriété **CardPropPps1Seq** comme défini au § 9 de la norme [ISO/CEI 7816-3]. Voir la NOTE.
CardPropPps1Seq	0x0014	Séquence d'un octet, d'une longueur maximale de 8, commençant par la valeur à établir de préférence pour PPS1 dans un échange PPS. Les valeurs sont définies au § 9.2 de la norme [ISO/CEI 7816-3], champ "T". Lecture et écriture.
CardPropInfdPref	0x0015	Un octet. La valeur indique la valeur préférée de l'IFSD à établir pour le protocole T=1 par le dispositif d'interface. Lecture et écriture.
RFU	autre	Réservé à une utilisation future.
NOTE – Les valeurs pour les octets PPS2 et PPS3 ne sont pas prises en charge par cette API, et ne doivent pas impérativement être prises en charge par l'**Hôte ECI**. Lecture et écriture.		

9.5.3.7 Codes d'erreur pour l'API de la Carte à puce

Les valeurs des erreurs propres à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.5.3.7-1.

Tableau 9.5.3.7-1 – Codes d'erreur de l'API de la Carte à puce

Nom	Valeur	Description
ErrCardOpenNot	-256	Aucune session de carte n'a été établie.
ErrCardConnFail	-257	Une session de carte a été établie, mais aucune connexion (après la remise à zéro).
RFU	**Autre**	Réservé à une utilisation future.

9.5.4 API d'acquisition de carrousel de données

9.5.4.1 Généralités

L'API d'acquisition de carrousel de données permet au **Client ECI** de récupérer des informations à partir d'un carrousel de radiodiffusion au format **ECI** comme défini au § 7.7.2. Un **Client ECI** peut l'utiliser notamment pour extraire des informations d'importation/exportation ayant pu être mises à jour.

NOTE – Les carrousels de données sont prévus pour transporter des données quasi statiques et ne constituent pas un protocole de transport de choix pour les données transitoires.

Un **Client ECI** peut lire les données d'un carrousel directement ou demander à l'**Hôte ECI** de surveiller les mises à jour d'un module ou groupe de carrousels qui l'intéressent. La surveillance peut s'effectuer à l'état PwrOn ou à des intervalles spécifiés pendant l'état de veille. Il est conseillé (pour des raisons de gestion de la consommation électrique) de faire coïncider ces périodes avec les périodes de surveillance de l'**Hôte ECI**.

L'**Hôte ECI** essaiera d'acquérir les données demandées et de les stocker dans un fichier en vue d'un accès ultérieur par le **Client ECI** par l'intermédiaire de l'API du système de fichiers. L'**Hôte ECI** garantit un nombre minimal de canaux d'acquisition parallèles par **Client ECI**, comme proposé dans le document [b-UIT-T J Suppl. 7].

Les messages de l'API d'acquisition de carrousel de données sont répertoriés dans le Tableau 9.5.4.1-1.

Tableau 9.5.4.1-1 – Message de l'API d'acquisition de carrousel de données

Message	Type	Sens	Étiquette	Description
reqDCAcqGroupInfo	A	C→H	0x0	Le **Client ECI** demande à l'**Hôte ECI** de lire la structure GroupInfoIndication dans le message DSI du carrousel de données ECI spécifié.
reqDCAcqModule	A	C→H	0x1	Le **Client ECI** demande à l'**Hôte ECI** d'acquérir un module de carrousel de données **ECI** particulier dans un fichier au moyen de paramètres de filtre de module et de divers modes.

9.5.4.2 Message reqDCAcqGroupInfo

C→H reqDCAcqGroupInfo (uint **operatorId,** uint **platformId)** →
H→C resDCAcqGroupInfo (byte **gii[])**

- Le **Client ECI** demande à l'**Hôte ECI** de lire la structure GroupInfoIndication dans le message DSI du carrousel de données **ECI** spécifié. Les codes d'erreur associés sont définis dans le Tableau 9.5.4.2-1.

Définitions des paramètres de la requête:

operatorId: uint	Identificateur composé de 20 bits de l'**Opérateur**, tel que présent dans la structure ECI_carousel_id transportée par le descripteur data_broadcast_id_descriptor() dans les PSI (voir § 7.7.2.4).
platformId: uint	Identificateur composé de 20 bits de l'**Opération de plate-forme**, tel que présent dans la structure ECI_carousel_id transportée par le descripteur data_broadcast_id_descriptor() dans les PSI (voir § 7.7.2.4).

Définitions des paramètres de la réponse:

gii: byte[]	Tableau d'octets comportant la structure GoupInfoIndication telle que transportée dans le message DSI du carrousel, comme défini pour la spécification DSM-CC appliquée à DVB [ETSI EN 301 192].

Sémantique détaillée:

- L'**Hôte ECI** ne fournit un accès qu'aux carrousels de clients qui sont chargés.

Tableau 9.5.4.2-1 – Codes d'erreur du message reqDCGroupInfo

Nom	Description
ErrDCAcqNetwAccessResource	Voir le Tableau 9.5.4.4-1.
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	

9.5.4.3 Message reqDCAcqModule

C→H reqDCAcqModule(uchar **aid,** fileName **fname**, uint **oId,** uint **pId,** byte **dType,** uint **model,** uint **version,** uint **index,** uint **mode)** →
H→C resDCAcqModule()

- Ce message autorise le **Client ECI** à demander à l'**Hôte ECI** d'acquérir un module de carrousel de données **ECI** particulier dans un fichier au moyen de paramètres de filtre de module et de divers modes.

Définitions des paramètres de la requête:

aid: uchar	Numéro du filtre d'acquisition. Un **Client ECI** ne peut disposer au maximum que de trois filtres d'acquisition actifs (valeurs 0 ... 2).
fname: fileName	Nom du fichier dans lequel les données du module de carrousel devant être acquises seront copiées. Les données existantes seront effacées.
oId: uint	Identificateur composé de 20 bits de l'**Opérateur**, tel que présent dans la structure ECI_carousel_id transportée par le descripteur data_broadcast_id_descriptor() dans les PSI (voir § 7.7.2.4).
pId: uint	Identificateur composé de 20 bits de l'**Opération de plate-forme**, tel que présent dans la structure ECI_carousel_id transportée par le descripteur data_broadcast_id_descriptor() dans les PSI (voir § 7.7.2.4).
dType: byte	Ce champ doit correspondre au champ de type de descripteur du groupe de modules, comme défini dans le Tableau 7.7.2.4-1.
model: uint	Transporte une valeur non signée composée de 16 bits, devant correspondre au champ de modèle de la structure compatibilityDescriptor du groupe à acquérir. Voir le Tableau 7.7.2.4-1.
version: uint	Transporte une valeur non signée composée de 16 bits, devant correspondre (filtre positif) ou non (filtre négatif), ou être ignorée lors de la mise en correspondance, par rapport au champ de version de la structure compatibilityDescriptor du groupe à acquérir, en fonction des bits 0 et 1 du paramètre **mode**. Voir le Tableau 7.7.2.4-1.
index: uint	Index du module auquel accéder dans le groupe. Ce paramètre doit être interprété en fonction du bit 1 du paramètre **mode**.

mode: uint	Ce paramètre comporte plusieurs champs: **bit 0:** indique un filtre positif ou négatif sur la **version**: 0b0 pour un filtre positif, 0b1 pour un filtre négatif; **bit 1:** indique si le filtre sur la **version** doit être ignoré (valeur 0b1) ou non (valeur 0b0); **bit 2:** indique si l'index doit être ignoré (valeur 1) et un module doit être acquis (pour les carrousels à module unique) ou si l'index doit être utilisé (modulo de numberOfModules, voir le Tableau 7.7.2.6-1); **bit 29:** si défini, l'**Hôte ECI** réalisera une acquisition en état de veille en vérifiant le carrousel conformément à ses propres exigences d'acquisition pour le carrousel en question, et cette acquisition continuera en mode veille et powerOn jusqu'à ce que les données demandées soient acquises; **bit 30:** indique si l'acquisition doit partir du principe que le carrousel de données est en cours d'exécution et que l'acquisition doit être réalisée pendant les heures planifiées normales du carrousel (valeur 0b0) ou si l'acquisition doit s'exécuter lorsque le carrousel peut être acquis et lorsque le filtre d'acquisition correspond (0b1) (c'est-à-dire, attendre que les données se présentent d'elles-mêmes); **bit 31:** activer (valeur 0b1) ou désactiver (valeur 0b0) l'acquisition avec cet **aid** de filtre.

Préconditions de la réponse:

1) Le module de carrousel demandé a été acquis, une erreur du système de fichier s'est produite ou, si le bit 30 de **mode** est défini, un problème d'acquisition est survenu.

2) L'**Hôte ECI** est à l'état PwerOn, c'est-à-dire que le **Client ECI** n'est pas mis en éveil lors d'une acquisition lorsqu'il est en état de veille.

Postconditions de la réponse:

1) Le fichier contient le module spécifié ou une erreur s'est produite.

2) Lorsque le bit 30 du paramètre **mode** est défini, aucune erreur d'acquisition ne peut survenir.

Sémantique détaillée:

- L'**Hôte ECI** ne fournit un accès qu'aux carrousels de **Clients ECI** qui sont chargés et pour lesquels il surveille le carrousel de données de radiodiffusion pour ses propres besoins.
- Si elle n'est pas configurée, l'acquisition en mode veille ne se fera pas. Les **Clients ECI** qui le souhaitent peuvent réaliser leur propre programmation d'acquisition au moyen de l'API de sortie de veille mentionnée au § 9.4.7.3.
- L'**Hôte ECI** fournira une **Réponse** "trivial" si une requête avec bit 31 du mode est validée.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.5.4.3-1.

Tableau 9.5.4.3-1 – Codes d'erreur du message reqDCAcqModule

Nom	Description
ErrDCAcqNetwAccessResource	Voir le Tableau 9.5.4.4-1.
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	
ErrDCAcqCarNoGroup	
ErrDCAcqCarNoModule	
ErrDCAcqCarTimeout	
ErrDCAcqFileSystemFailure	
ErrDCAcqFileQuotaExceeded	

9.5.4.4 Codes d'erreur de l'API d'acquisition de carrousel de données

Les valeurs des erreurs propres à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.5.4.4-1.

Tableau 9.5.4.4-1 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrDCAcqNetwAccessResource	-256	Voir le Tableau 9.6.2.3.7-1.
ErrDCAcqNetwAccessFail	-257	Voir le Tableau 9.6.2.3.7-1.
ErrDCAcqNoCarousel	-258	Aucun carrousel comportant un identificateur d'**Opérateur** ou d'**Opération de plate-forme** correspondant n'a été trouvé sur les réseaux de radiodiffusion accessibles à l'**Hôte ECI**.
ErrDCAcqCarNoGroup	-260	La structure groupInfoIndication a été trouvée dans le message DSI du carrousel mais aucun groupe correspondant n'a été trouvé.
ErrDCAcqCarNoModule	-261	Le groupe du carrousel (DII) a été trouvé, mais aucun module correspondant n'a été trouvé.
ErrDCAcqCarTimeout	-262	Le délai d'attente a expiré lors de l'accès au message DSI, DII ou DDB du carrousel.
ErrDCAcqFileSystemFailure	-263	Voir le Tableau 9.4.5.5-1.
ErrDCAcqFileQuotaExceeded	-264	Voir le Tableau 9.4.5.5-1.

9.6 API d'accès à la ressource de déchiffrement de l'Hôte ECI

9.6.1 API de déchiffrement de l'Hôte ECI

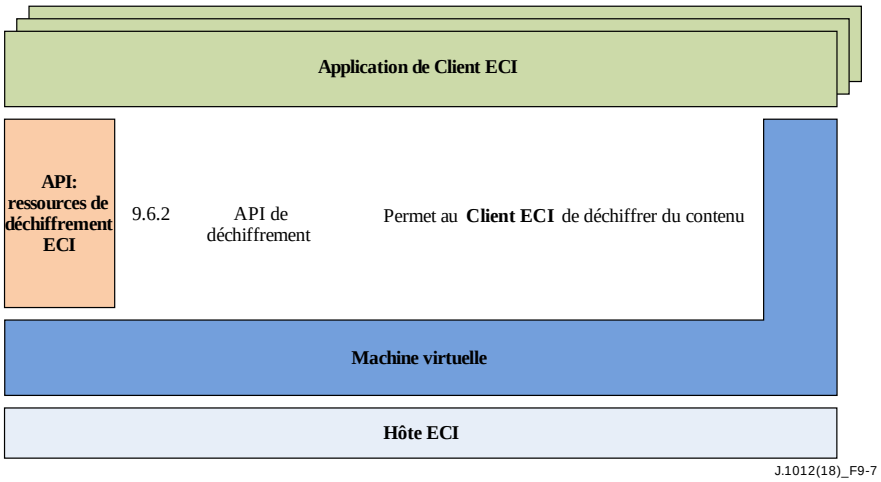


Figure 9.6.1-1 – Représentation des API définies dans le § 9.6

Le Tableau 9.6.1-1 répertorie les API présentées dans le § 9.6 et la Figure 9.7.1 illustre leur positionnement dans l'architecture **ECI**.

Tableau 9.6.1-1 – Liste des API définies dans le § 9.6

Paragraphe	Nom de l'API	Description
9.6.2	API de déchiffrement de l'**Hôte ECI**	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** des informations URI standard liées à un certain élément de contenu.

9.6.2 Définition de l'API de déchiffrement de l'Hôte ECI

9.6.2.1 Introduction

Les API de déchiffrement permettent à l'**Hôte ECI** (par exemple sur demande d'applications résidentes ou téléchargées) de sélectionner un **Client ECI** correspondant aux exigences de déchiffrement du contenu et de demander à le déchiffrer. Tous les messages de déchiffrement entre le **Client ECI** et l'**Hôte ECI** sont échangés dans le cadre d'un **Pointeur de média** qui représente le contenu, de réseaux de transmission associés et des ressources nécessaires pour le décoder.

Les API de déchiffrement sont les suivantes:

1) API générique de session de média pour tous les types de média, avec notamment une fonctionnalité de mise en correspondance entre le contenu et le **Client ECI**.

2) API de déchiffrement du flux de transport

3) API de déchiffrement de fichier et de flux

9.6.2.2 API de session de média

9.6.2.2.1 Généralités

Le **Client ECI** peut annoncer la liste des spécificateurs de correspondance selon lesquels l'**Hôte ECI** peut le mettre en correspondance avec le contenu.

L'**Hôte ECI** peut demander à un **Client ECI** correspondant d'ouvrir une session de désembrouillage pour un **Pointeur de média**. L'ouverture d'une session n'implique pas le commencement du décodage. Elle vise simplement à garantir que les ressources nécessaires pour accéder au contenu et/ou à ses métadonnées et pour effectuer une session de désembrouillage sont disponibles à la fois côté **Hôte ECI** et côté **Client ECI**. Avant de confirmer une session, les **Clients ECI** doivent s'assurer que les **Cartes à puce** ou autres ressources nécessaires au désembrouillage du contenu sont disponibles. Le Tableau 9.6.2.2.1-1 répertorie les fonctions de l'API.

Tableau 9.6.2.2.1-1 – Messages de l'API de session de déchiffrement d'un Pointeur de média

Message	Type	Sens	Étiquette	Description
setDcrMhMatch	set	C→H	0x0	Indique à l'**Hôte ECI** sous quels identificateurs le **Client ECI** peut être reconnu pour le désembrouillage du contenu.
reqDcrMhOpen	A	H→C	0x1	L'**Hôte ECI** demande au **Client ECI** d'ouvrir une session de média d'un type spécifié au moyen d'un **Pointeur de média**.
reqDcrMhClose	A	H→C	0x2	L'**Hôte ECI** ferme une session de média avec un **Client ECI**.
reqDcrMhBcAlloc	A	C→H	0x3	Le **Client ECI** demande un **Pointeur de média** pour ses propres besoins d'accès à un réseau de radiodiffusion.
reqDcrMhCancel	A	C→H	0x4	Le **Client ECI** annule une session de média avec l'**Hôte ECI**.

9.6.2.2.2 Message d'API setDcrMhMatch

C→H setDcrMhMatch(uint **matchListLength,** MatchSpecifier **matchList[])**

- Ce message autorise le **Client ECI** à indiquer à l'**Hôte ECI** les identificateurs de système de déchiffrement pour lesquels il est capable de fournir des services de déchiffrement de flux de transport.

NOTE – La capacité effective à déchiffrer le contenu peut dépendre de l'abonnement, du statut de paiement ou d'autres conditions.

Définition de la propriété SetDcrMhMatch

matchListLength: uint	Longueur de la liste de correspondance (matchList) en ce qui concerne les spécificateurs.
matchList: MatchSpecifier[].	Tableau 9.6.2.2.2-1. L'**Hôte ECI** utilisera cette liste pour mettre en correspondance le contenu avec les capacités de déchiffrement potentielles du **Client ECI**, conformément au § 9.5.3.3. Les spécificateurs de correspondance sont définis par le type MatchSpecifier. Tous les champs de MatchSpecifier doivent correspondre au contenu pour qu'une correspondance soit générée.

Tableau 9.6.2.2.2-1 – Définitions du type MatchSpecifier

```
#define MaxMhSubFormat 16;
typedef struct MatchSpecifier {
   uchar decryptIdType; /*voir le Tableau 9.6.2.2.2-2
*/
   union decryptId {
      bool     ECI Client ID;
      ushort   dvbCaId;
      byte     uuid[16];
   }
   byte  mhType;
   byte  subFormat[MaxMhSubFormat];
} MatchSpecifier;
```

Tableau 9.6.2.2.2-2 – Définition du champ decryptIdType du message setDcrMhMatch

Nom	Valeur	Description
None	0x00	Ne correspond à aucun contenu d'une **Requête** émise; renvoie "no match" en cas d'ouverture d'une session.
ClientEciId	0x01	L'identification du **Client ECI** peut se faire sur la base de l'identificateur du **Client ECI**, composé des valeurs de 20 bits (champs de type et de version exclus) <<operator_id,platform_operation_id>,<vendor_id,client_id>> spécifiées au § 7 de la présente Recommandation.
ClientDvbCaId	0x02	decryptId est un identificateur de système d'accès conditionnel tel que défini dans les normes [CEN EN 50221] et [ETSI EN 301 192]. Cette valeur indique que dvbCaId est la variante utilisée pour l'union specifierType. Les valeurs réelles sont telles que définies dans la norme [CEN EN 50221].
ClientUUID	0x03	L'élément déchiffré est un identificateur DRM tel que défini par les standards CENC/DASH, spécifié sous forme d'UUID [IETF RFC 4122].
RFU	autre	Réservé à une utilisation future.

mhType: uint	Type du **Pointeur de média** (mode de déchiffrement principal) pris en charge par le **Client ECI** pour cet identificateur de client ECI (ClientEciID).
subFormat: byte[]	Ce paramètre permet de définir des spécifications de type supplémentaires pour le **Client ECI**. L'interprétation de ces octets dépend de **mhType** tel que défini dans le Tableau 9.6.2.2.2-3

Tableau 9.6.2.2.2-3 – Définition de type de subFormat

Valeur de mhType	Sémantique du champ subFormat
ISOBMFF	Le champ subFormat contient zéro ou davantage de définitions séquentielles FourCC des valeurs de marques de la boîte ftyp ou styp ISOBMFF pouvant être décodées par le **Client ECI**. Une (ou davantage) de ces valeurs FourCC doit correspondre aux valeurs `major_brand` ou `compatible_brands[]` de la boîte ftyp ou styp du conteneur ISOBMFF. La valeur 0x0000 dans le champ subFormat signifie aucune valeur (ne jamais définir de correspondance), la valeur 0xFFFF comme première entrée signifie toute valeur de marque (quels que soient les octets suivants).
Autre	Réservé à une utilisation future.

Sémantique détaillée:

Lorsqu'il tente de rendre le contenu basé sur le flux de transport, l'**Hôte ECI** doit essayer de le mettre en correspondance avec les **Clients ECI** disponibles en suivant les règles suivantes, par ordre de priorité:

1) L'**Hôte ECI** essaiera d'établir un ensemble de spécificateurs de correspondance applicables en utilisant les identificateurs des **Clients ECI** pour ce contenu, comme défini au § 7.2.2. Si

un identificateur de **Client ECI** applicable et les propriétés de correspondance associées correspondent au **MatchSpecifier** d'un **Client ECI**, l'**Hôte ECI** proposera du contenu à déchiffrer à ce Client. Si plusieurs **Clients ECI** correspondent, l'**Hôte ECI** appliquera la procédure suivante:

a) L'**Hôte ECI** proposera le contenu pour déchiffrement au **Client ECI** ayant fourni le plus récemment avec succès les mots de contrôle destinés au déchiffrement de contenus issus de la même "source".

b) Si le premier **Client ECI** ne parvient pas à déchiffrer le contenu, l'**Hôte ECI** s'efforcera d'utiliser d'autres **Clients ECI** correspondant, en commençant par celui ayant effectué le plus récemment avec succès un déchiffrement de contenu issu de la "source".

2) Si l'**Hôte ECI** ne parvient pas à trouver d'identificateur de **Client ECI** correspondant au contenu, ou si aucun des **Clients ECI** identifiés conformément au point 1) ci-dessus ne parvient à décoder le contenu, l'**Hôte ECI** s'efforcera d'établir un autre ensemble d'identificateurs pour le contenu, comme défini au § 9.5.4.3. Si un seul identificateur et les propriétés de correspondance associées correspondent à un **Client ECI**, l'**Hôte ECI** proposera le contenu à déchiffrer à ce **Client ECI**. Si plusieurs **Clients ECI** correspondent, l'**Hôte ECI** appliquera la procédure suivante:

a) L'**Hôte ECI** proposera le contenu pour déchiffrement au **Client ECI** ayant déchiffré le plus récemment avec succès des contenus issus de la même "source".

b) Si le premier **Client ECI** ne parvient pas à déchiffrer le contenu, l'**Hôte ECI** s'efforcera d'utiliser d'autres **Clients ECI** correspondant, en commençant par celui ayant effectué le plus récemment avec succès un déchiffrement de contenu issu de la "source".

Le terme "source" employé ci-dessus doit comprendre au minimum:

1) Un réseau de radiodiffusion DVB, ou un bouquet d'un tel réseau, fournissant le flux de transport.

2) Un site Internet utilisé pour la navigation avec un navigateur proposant des références vers du contenu.

9.6.2.2.3 Message reqDcrMhOpen

H→C reqDcrMhOpen(ushort **mH,** MatchSpecifier **match) →**
C→H resDcrMhOpen(ushort **mH)**

- Ce message permet à l'**Hôte ECI** de demander une session de déchiffrement avec le **Client ECI**. Le **Client ECI** doit réserver toutes les ressources normalement requises pour effectuer le déchiffrement, telles qu'identifiées dans les paramètres **mh** et **match**. Les codes d'erreur associés sont définis dans le Tableau 9.6.2.2.3-1.

Définition des paramètres de la requête

mH: ushort	**Pointeur de média** du contenu à déchiffrer.
match: MatchSpecifier	Copie du spécificateur correspondant (contient également le type de pointeur de média de la session).

Définition des paramètres de la réponse

mH: ushort	**Pointeur de média** du contenu à déchiffrer.

Préconditions de la requête:

- L'**Hôte ECI** a réservé toutes les ressources requises pour déchiffrer le contenu. Pour les contenus de flux de transport, cela inclut tous les réglages ou autres ressources d'accès au réseau et les contrôles applicables en conséquence, les ressources de démultiplexage et les ressources de désembrouillage pour une application de paire de mots de contrôle au moins.

Postconditions de la réponse:

- En cas de réussite, le **Client ECI** a réservé toutes les ressources généralement nécessaires pour décoder du contenu pour la session demandée, y compris l'accès à toutes les ressources externes (serveurs DRM, **Cartes à puce**, etc.) requises en principe pour une opération de déchiffrement.

NOTE – Les ressources nécessaires à titre exceptionnel ou les ressources pouvant normalement être obtenues à la demande sont exclues.

- Si l'erreur ErrDcrUserDelay est renvoyée, le **Client ECI** attend une action de la part de l'**Utilisateur** pour pouvoir ouvrir la session (par exemple pour accéder à une **Carte à puce**). L'**Hôte ECI** doit continuer d'envoyer la requête reqDcrMhOpen (avec les mêmes paramètres) jusqu'à ce qu'un résultat positif ou une erreur définitive soit renvoyé, ou peut envoyer une requête reqDcrMhClose pour terminer la session en cours. Le **Client ECI** peut annuler la demande au moyen du message reqDcrMhCancel s'il ne parvient pas à obtenir la réponse requise de la part de l'**Utilisateur**.

Tableau 9.6.2.2.3-1 – Codes d'erreur du message reqDcrMhOpen

Nom	Description
ErrDcrUserDelay	Voir le Tableau 9.6.2.2.7-1.
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.2.4 Message reqDcrMhClose

H→C reqDcrMhClose(ushort **mH) →**
C→H resDcrMhClose(ushort **mH)**

- Ce message permet à l'**Hôte ECI** de fermer une session de déchiffrement avec le **Client ECI**. Le **Client ECI** peut libérer les ressources utilisées pour cette session.

Définition des paramètres de la requête

mH: ushort	**Pointeur de média** de la session à fermer.

Définition des paramètres de la réponse

mH: ushort	**Pointeur de média** de la session fermée.

Postconditions de la requête:

- Le **Client ECI** libère toutes les ressources dont il a eu besoin en particulier pour la session.

Postconditions de la réponse:

- L'**Hôte ECI** peut libérer toute ressource liée au **Pointeur de média**.

9.6.2.2.5 Message reqDcrMhBcAlloc

C→H reqDcrMhBcAlloc(byte **networkType[2],** uchar **priority,** char **reason[80]) →**
H→C resDcrMhBcAlloc(ushort **mH)**

- Ce message permet au **Client ECI** de demander une connexion à un réseau de radiodiffusion dans le but d'acquérir des données de sécurité.

Définition des paramètres de la requête

networkType: byte[2]	Type de réseau de radiodiffusion auquel le client ECI veut accéder; voir le Tableau 9.6.2.3.6.2-3 pour les valeurs.
priority: uchar	La priorité pour l'accès au réseau est définie dans le Tableau 9.6.2.2.5-1.
reason: char[80]	Chaîne de 80 caractères au maximum et terminée par une valeur "null" pouvant être présentée à l'**Utilisateur** pour résoudre un conflit de ressources au sein de l'hôte ECI afin de satisfaire à cette requête.

Tableau 9.6.2.2.5-1 – Définition de la priorité d'accès au réseau de radiodiffusion

Nom	Valeur	Description
DcrAllocPrioBackground	0x01	L'accès est requis pour un traitement en arrière-plan; il peut être refusé ou interrompu si une tâche présentant une priorité plus élevée a besoin d'accéder aux ressources: par exemple, l'accès aux messages EMM ou aux données relatives à la capacité de renouvellement de la sécurité sur un multiplex central.
DcrAllocPrioActivec	0x02	L'accès est requis pour une fonction de désembrouillage principale et peut occasionner une gêne pour l'**Utilisateur** s'il n'est pas accordé (ou est interrompu): Par exemple, une session de visionnement demandée par l'**Utilisateur** ou une session d'enregistrement programmée au préalable par l'**Utilisateur**.
RFU	autre	Réservé à une utilisation future.

Définition des paramètres de la requête

mH: ushort	**Pointeur de média** de la session ouverte.

Sémantique détaillée:

- L'**Hôte ECI** peut annuler la session en utilisant le message reqDcrMhClose si une autre tâche à la priorité plus élevée requiert des ressources d'accès au réseau.
- Le **Client ECI** fermera la session au moyen du message reqDcrMhCancel s'il n'a plus besoin d'accéder au réseau.

Postconditions de la requête:

1) L'**Hôte ECI** a attribué toutes les ressources nécessaires pour accéder au type de réseau demandé.

Postconditions de la réponse:

1) Le **Client ECI** doit effectuer un réglage en vue d'acquérir un flux de transport à l'aide du message reqDcrTsRelocate avant de commencer l'acquisition d'une section.

Tableau 9.6.2.2.5-2 – Codes d'erreur du message reqDcrMhBcAlloc

Nom	Description
ErrDcrNetworkAccessCapability	Voir le Tableau 9.6.2.2.7-1.
ErrDcrNetworkAccessResource	
ErrDcrPrioOverride	
ErrDcrResourceMissing	

9.6.2.2.6 Message reqDcrMhCancel

C→H reqDcrMhCancel(ushort **mH,** uchar **reason) →**
H→C resDcrMhCancel(ushort **mH)**

- Ce message permet au **Client ECI** de fermer une session de déchiffrement avec l'**Hôte ECI**. Le **Client ECI** a libéré toutes les ressources nécessaires en particulier pour cette session.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** de la session à fermer.
reason: uchar	Raison de l'annulation de la session de déchiffrement. Les valeurs sont définies dans le Tableau 9.6.2.2.6–1.

Tableau 9.6.2.2.6-1 – Valeurs des raisons du message reqDcrMhCancel

Nom	Valeur	Description
DrcMhUndefined	0x00	Une erreur inconnue s'est produite sur le **Client ECI** et l'a obligé à fermer la session.
DcrMhCardMissing	0x01	Une **Carte à puce** est nécessaire au décodage mais n'a pu être (re-)connectée pour faciliter le déchiffrement du contenu dans un délai raisonnable.
DcrMhServiceMissing	0x02	Un service (externe à l'**Équipement CPE**) aidant le **Client ECI** à fournir les services de déchiffrement nécessaires au maintien de la session de déchiffrement n'est pas disponible dans un délai raisonnable.
DcrMhResourceMissing	0x03	Une ressource (interne à l'**Équipement CPE**) nécessaire pour fournir les services de déchiffrement n'est pas disponible pour le **Client ECI** dans un délai raisonnable (à l'exclusion de **DcrMhMmiMissing**).
DcrMhMmiMissing	0x04	Le **Client ECI** n'est pas parvenu à atteindre une ressource de session MMI destinée à l'interaction avec l'**Utilisateur** et requise pour maintenir la session de déchiffrement dans un délai raisonnable.
DcrMhAllocTerminate	0x05	Un **Pointeur de média** a été attribué pour le compte du **Client ECI** par l'intermédiaire de reqDcrMhBcAlloc et n'est plus requis par le **Client ECI**.
RFU	Autre	Réservé à une utilisation future.

Un délai raisonnable pour l'annulation d'une session de **Pointeur de média** par l'**Hôte ECI** est proposé dans le document [b-UIT-T J Suppl. 7].

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** de la session annulée.

Préconditions de la requête:

- Le **Client ECI** a libéré les ressources dont il a eu besoin en particulier pour la session.

Postconditions de la requête:

- L'**Hôte ECI** peut libérer toute ressource liée au **Pointeur de média**.

Postconditions de la réponse:

- L'Hôte ECI ferme la session de Pointeur de média.

9.6.2.2.7 Codes d'erreur de l'API de session de média

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.6.2.2.7-1.

Tableau 9.6.2.2.7-1 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrDcrUserDelay	-256	Le système a attendu longtemps une action de la part de l'**Utilisateur** nécessaire pour réaliser l'opération. L'opération n'a pas abouti.
ErrDcrCardMissing	-257	La **Carte à puce** requise pour la session n'est pas accessible/disponible.
ErrDcrServiceMissing	-258	Un service externe à l'**Équipement CPE** requis pour aider le **Client ECI** dans les opérations de déchiffrement n'est pas disponible.
ErrDcrResourceMissing	-259	Une ressource non définie interne à l'**Équipement CPE** nécessaire pour accéder au contenu ou le déchiffrer n'est pas disponible.
ErrDcrMmiMissing	-260	L'accès du **Client ECI** à l'interface homme-machine n'est pas disponible.
ErrDcrDescrContinue	-261	L'**Hôte ECI** continue à tenter de désembrouiller le contenu de ce flux de transport.
ErrDcrNetworkAccessCapability	-262	Il manque à l'**Hôte ECI** une ressource d'accès au réseau pour localiser le flux de transport requis.
ErrDcrNetworkAccessResource	-263	L'**Hôte ECI** ne peut acquérir la ressource d'accès au réseau nécessaire pour accéder au flux de transport demandé.
ErrDcrPrioOverride	-264	Une tâche de priorité plus élevée dans l'**Équipement CPE** a eu besoin des ressources pour le **Pointeur de média** et a terminé en conséquence la session de **Pointeur de média**.
RFU	autre	Réservé à une utilisation future.

9.6.2.3 Désembrouillage des données des flux de transport

9.6.2.3.1 Introduction

L'**Hôte ECI** peut demander au **Client ECI** d'effectuer une session de désembrouillage (d'un type spécifique, ici le type de radiodiffusion MPEG) en lui fournissant un **Pointeur de média** (voir le § 9.1.2). L'**Hôte ECI** fournira les données de sécurité, tel que spécifié par le **Client ECI** pour désembrouiller les données.

Pour le désembrouillage de contenus dans la plupart des formats de flux de transport, l'interface **ECI** utilise un modèle de synchronisation implicite pour synchroniser les mots de contrôle avec le contenu proposé au désembrouilleur. Dans ce modèle, l'**Hôte ECI** fournit au **Client ECI** les données de contrôle de sécurité issues du flux de transport à mesure qu'il est démultiplexé et désembrouillé. Le **Client ECI** fournit les mots de contrôle requis (généralement deux par flux élémentaire, souvent identiques pour tous les flux élémentaires) au moment adéquat. Le **Client ECI** décode généralement un message de commande d'habilitation (ECM) pour en extraire les mots de contrôle et charge immédiatement ces derniers dans le désembrouilleur. L'application de ces mots de contrôle est synchronisée avec le flux par l'intermédiaire de la signalisation du flux de contenu, au moyen des bits de commande d'embrouillage au niveau des paquets TS ou PES.

L'API est décrite dans les paragraphes suivants:

1) Démarrage, redémarrage et arrêt d'un déchiffrement de flux de transport (§ 9.6.2.3).

2) Acquisition de données de sécurité (§ 9.6.2.3.5).

3) Fonctions de syntonisation de la radiodiffusion (§ 9.6.2.3.6).

9.6.2.3.2 Format du flux de transport et versions de session

Les flux de transport désembrouillés par l'intermédiaire d'un **Pointeur de média** avec le type de session de média **MhDvbTsBroadcast** doivent être conformes aux spécifications suivantes: [ISO/CEI 13818-1] (notamment l'application des bits de commande d'embrouillage aux paquets TS) et [ETSI ETR 289].

9.6.2.3.3 Exigences de traitement de l'Hôte ECI

9.6.2.3.3.1 Détection du chiffrement de l'embrouillage

L'**Hôte ECI** signalera au **Client ECI** le mode de chiffrement applicable en se basant sur les règles suivantes:

1) Pour les flux DVB, il emploiera la signalisation indiquée par le descripteur d'embrouillage contenu dans la table de correspondance du programme (PMT), comme défini dans les normes [ETSI TS 103 127] et [ETSI TS 100 289].

2) Si aucun descripteur n'est trouvé conformément au critère 1) et que la source est un réseau de radiodiffusion DVB, l'**Hôte ECI** partira du principe que le mode CSA1 est utilisé comme spécifié dans la définition du descripteur d'embrouillage.

9.6.2.3.3.2 Détection de l'identificateur CA

Afin d'établir la liste des identificateurs CA DVB applicables pour un service embrouillé, l'embrouillage étant détecté par les bits d'embrouillage des paquets TS ou PES, l'**Hôte ECI** utilisera la séquence suivante de règles d'acquisition dans un flux de transport (issu d'un réseau de radiodiffusion ou autre):

1) Il s'efforcera d'extraire les descripteurs de CA (CA_descriptors) transportés par la PMT du service. En cas d'échec et si le contenu est embrouillé.

2) Il s'efforcera d'extraire les champs CA_system_id du descripteur de l'identificateur de CA transporté par un bouquet DVB, une table de description des services (SDT) ou une table d'informations d'événement (EIT) applicable pour le contenu.

NOTE – Pour certaines sources de contenu basé sur flux de transport, l'identificateur CA ou DRM applicable peut être identifié par d'autres moyens.

9.6.2.3.4 Démarrer et arrêter le déchiffrement du flux de transport

9.6.2.3.4.1 Généralités

L'**Hôte ECI** peut lancer le déchiffrement du contenu sur un **Pointeur de média** ouvert en utilisant les ressources du **Client ECI** réservées. L'**Hôte ECI** fournira une table "CA-PMT" contenant la spécification des flux élémentaires à déchiffrer. Le Tableau 9.6.2.3.4.1–1 répertorie les messages de l'API de déchiffrement disponibles.

Tableau 9.6.2.3.4.1–1 – API de déchiffrement pour le contenu du flux de transport du Pointeur de média

Message	Type	Sens	Étiquette	Description
reqDcrTsDescrStart	A	H→C	0x08	Demande au **Client ECI** de désembrouiller un programme dans un flux de transport ou de renvoyer son statut de désembrouillage.
reqDcrTsDescrStop	A	H→C	0x09	L'**Hôte ECI** demande au **Client ECI** de désembrouiller un **Pointeur de média**.
reqDcrTsDescrQuit	A	C→H	0x0A	Le **Client ECI** termine une session de désembrouillage avec l'**Hôte ECI**.

9.6.2.3.4.2 Message reqDcrTsDescrStart

H→C reqDcrTsDescrStart(ushort **mH,** uint **caPmtLen,** byte **caPmt[]) →**
C→H resDcrTsDescrStart(ushort **mH,** unit **sizeofEsStat,** descrStat **esStat[])**

- Ce message permet au **Client ECI** de démarrer le déchiffrement d'un programme défini par le paramètre **caPmt** dans le flux identifié par le paramètre **mH**, ou demande quelles sont les capacités ou conditions pour ce faire.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
caPmtLen: uint	Longueur en octets du paramètre **caPmt**.
caPmt: byte[]	L'objet ca_pmt est défini au § 8.4.3 de la norme [CEN EN 50221] dans l'ordre des octets du réseau, l'interprétation des paramètres ca_pmt_list_management et ca_pmt_cmd_id étant modifiée conformément au Tableau 9.6.2.3.4.2–1.

Les valeurs et la sémantique du paramètre ca_pmt_list_management seront conformes aux définitions du Tableau 9.6.2.3.4.2–1.

Tableau 9.6.2.3.4.2-1 – Valeurs de ca_pmt_list_management

Nom	Valeur	Description
DcrTsDescrStartOnly	0x03	Un programme unique doit être désembrouillé dans le service. Il peut s'agir d'une valeur nouvelle ou mise à jour.
DcrTsDescrStartUpdate	0x05	Même signification que **DcrTsDescrStartOnly**.
RFU	Autre	Réservé à une utilisation future.

Les valeurs du paramètre ca_pmt_cmd_id seront identiques à celles indiquées au § 8.4.3 de la norme [CEN EN 50221], avec les restrictions suivantes:

1) La valeur 0x02 (ok_mmi) n'est pas autorisée.

2) Les valeurs 0x01 (ok_descrambling) et 0x03 (query) ne doivent pas figurer dans la même structure ca_pmt. Autrement dit, une **Requête** doit, soit être une interrogation pure, soit une requête de désembrouillage pure.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.
sizeofEsStat: uint	Nombre d'octets du paramètre esStat.
esStat: descrStat	Statut de désembrouillage des flux élémentaires spécifiés dans le paramètre **caPmt** de la **Requête**. descrStat est défini dans le Tableau 9.6.2.3.4.2–2. Une valeur **descrStat.pid** ne figurera qu'une seule fois dans le paramètre **esStat**. Chaque paramètre elementary_PID de la structure ca_pmt définie dans la norme [CEN EN 50221] figurera une fois sauf si son paramètre ca_pmt_cmd_id correspondant porte la valeur 0x04 (not_selected), auquel cas il ne figurera pas dans le paramètre esStat.

Tableau 9.6.2.3.4.2–2 – Définition de type pour la structure descrStat

```
typedef struct descrStat {
   ushort pid;
   uchar    caStatus
} descrStat;
```

pid: ushort	Valeur du PID du flux à désembrouiller.
caStatus: uchar	Les valeurs doivent correspondre à la définition du paramètre CA_enable de l'objet ca_pmt_reply telle qu'indiquée au § 8.4.3 de la norme [CEN EN 50221].

Sémantique détaillée:

1) L'**Hôte ECI** émettra cette commande si l'ensemble de flux élémentaires à décoder doit changer.

2) L'**Hôte ECI** émettra une **Requête reqDcrTsDescrEnd** si la session de média est arrêtée. Dans le cas contraire, le **Client ECI** pourrait être indûment amené à croire que l'**Utilisateur** continue de visionner le contenu et des frais pourraient être appliqués en conséquence.

3) Les codes d'erreur associés sont définis dans le Tableau 9.6.2.3.4.2–3.

Préconditions de la requête:

1) Le **Pointeur de média** est ouvert et au format TS.

Postconditions de la requête:

1) Le **Client ECI** peut démarrer les actions de désembrouillage et utiliser d'autres fonctions liées au flux de transport du **Pointeur de média**.

Tableau 9.6.2.3.4.2-3 – Codes d'erreur du message reqDcrTsDescrStart

Nom	Description
ErrDcrUserDelay	Voir le Tableau 9.6.2.3.7–1.
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.3.4.3 Message reqDcrTsDescrStop

H→C reqDcrTsDescrStop(ushort **mH) →**
C→H resDcrDescrStop(ushort **mH)**

- Ce message permet à l'**Hôte ECI** d'indiquer au **Client ECI** qu'il doit arrêter l'opération de désembrouillage du flux de transport liée au **Pointeur de média mH** actuel.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Préconditions de la réponse:

1) Toute opération du **Client ECI** liée au désembrouillage du **Pointeur de média** mH est terminée.

9.6.2.3.4.4 Message reqDcrTsDescrQuit

C→H reqDcrTsDescrQuit(ushort **mH, ushort** reason**) →**
H→C resDcrDescrQuit(ushort **mH)**

- Ce message permet au **Client ECI** d'informer l'**Hôte ECI** qu'il a arrêté de traiter les clés aux fins de l'opération de désembrouillage du flux de transport liée au **Pointeur de média mH** actuel.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
reason: ushort	Raison pour laquelle le **Client ECI** a terminé le traitement des clés aux fins de l'opération de désembrouillage, comme défini dans le Tableau 9.7.2.5.9–1.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Préconditions de la réponse:

1) Toutes les activités de l'**Hôte ECI** liées au désembrouillage du **Pointeur de média mH** se sont terminées ou une erreur a été renvoyée.

Postconditions de la réponse:

1) Toute activité du **Client ECI** liée au **Pointeur de média mH** prend fin immédiatement ou une erreur est renvoyée.

Tableau 9.6.2.3.4.4-1 – Codes d'erreur du message reqDcrTsDescrQuit

Nom	Description
ErrDcrDescrContinue	Voir le Tableau 9.6.2.3.7-1.

9.6.2.3.5 Acquisition de données de déchiffrement dans le flux de transport par le Client ECI

9.6.2.3.5.1 Généralités

Le **Client ECI** peut acquérir des données TS intrabande nécessaires au déchiffrement sous forme de sections du flux de transport associé au **Pointeur de média**. Le procédé le plus simple à cet effet consiste à définir un filtre de section. Pour accélérer l'acquisition lors d'un changement de chaîne, le **Client ECI** peut définir un filtre de section par défaut incluant la PMT et le flux ECM. Il peut également lire d'autres tables MPEG et DVB standard à partir de l'**Hôte ECI**. Les sections MPEG sont des structures de données telles que définies au § 2.4.4.11 de la norme [ISO/CEI 13818-1], structure private_section(). Les fonctions de cette partie de l'API de flux de transport MPEG sont répertoriées dans le Tableau 9.6.2.3.5.1-1.

Tableau 9.6.2.3.5.1-1 – Messages de contrôle du désembrouillage du flux de transport par l'Hôte ECI

Message	Type	Sens	Étiquette	Description
setDcrTsSectionAcqDefault	set	C→H	0x10	Définit un filtre par défaut pour l'acquisition de section.
setDcrTsSectionAcq	set	C→H	0x11	Définit un filtre pour les acquisitions de section.
reqDcrTsSection	A	H→C	0x12	Transfert une section acquise au **Client ECI**.
reqDcrTsTable	A	C→H	0x13	Le **Client ECI** acquiert une table dans le flux.

9.6.2.3.5.2 Spécification du filtre de section

Les sections MPEG telles que définies au § 2.4.4.11 de la norme [ISO/CEI 13818-1] peuvent être extraites d'un flux de transport selon une spécification communiquée par un **Client ECI** à l'**Hôte ECI**. Un **Hôte ECI** prendra en charge huit filtres de section pour un **Client ECI**. Une configuration de filtre de section permet au **Client ECI** de réaliser un filtre à partir d'un PID du flux de transport avec un nombre limité de spécificateurs indirects (par exemple pour la PMT). Elle permet au **Client ECI** de définir des filtres positifs (les champs de section sélectionnés satisfont à la spécification du **Client ECI**) et des filtres négatifs (les données de section diffèrent de la spécification de filtre du **Client ECI**). Les sections filtrées peuvent être regroupées et envoyées soit lorsque la taille maximale du tampon est atteinte, soit dès qu'elles sont nécessaires.

Le filtrage des octets de section ignorera les deuxième et troisième octets d'une section.

La spécification pour un filtre de section est indiquée dans le Tableau 9.6.2.3.5.2-1.

Tableau 9.6.2.3.5.2-1 – Définition du type DcrSectionFilterSpec structure#define DcrSectionFilterMaxlen 16

```
#define DcrSectionFilterMaxlen 16
typedef struct dcrSectionFilterSpec {
   ushort    pid;
   ushort    caId;
   ushort    bufferSize;
   uint      timeout;
   uint      modeFlags;
   byte      filter[DcrSectionFilterMaxlen];
   byte      mask[DcrSectionFilterMaxlen];
   byte      neg[DcrSectionFilterMaxlen];
} dcrSectionFilterSpec;
```

La sémantique est la suivante:

pid: ushort	PID des paquets TS à filtrer. Les valeurs des PID doivent être représentées par leur valeur non signée sur 13 bits, soit entre 0x0000 et 0x1FFF. Le PID de la PMT du flux à acquérir est représenté par la valeur 0x8000. Le PID d'un flux ECM associé à acquérir est représenté par la valeur 0x8001.
cald: ushort	Ce champ n'est pertinent que lorsque le champ **pid** porte la valeur 0x8001. Dans ce cas, la valeur de ce champ est l'identificateur CA MPEG/DVB du système d'accès conditionnel pour lequel le flux ECM sera acquis. L'**Hôte ECI** analysera la PMT du service à désembrouiller et mettra en correspondance le champ **cald** avec les descripteurs de CA (CA_descriptors) (définis dans la norme [ISO/CEI 13818-1]) applicables au PID de la vidéo s'il y en a ou avec le premier flux élémentaire de la PMT, et utilisera le champ CA-PID du descripteur correspondant pour identifier le flux ECM à acquérir et filtrer.
bufferSize: ushort	Taille maximale du tampon. Une section au moins sera mise en mémoire tampon. Si ce champ est défini sur zéro, chaque section sera transférée séparément.
timeout: uint	Délai d'expiration en ms pour le filtrage d'une section unique. Redémarre à chaque section filtrée avec succès. Si ce champ est défini sur zéro, il n'y a pas de délai d'expiration.
modeFlags: uint	Si le bit 0 est défini, l'**Hôte ECI** évitera d'envoyer deux fois la même section au **Client ECI**. A cet effet, il utilisera un tampon de sections acquises précédemment à hauteur de 64 ko au maximum. Tous les autres bits sont réservés et seront mis à 0 par le **Client ECI**.
filter: byte []	Valeur à mettre en correspondance avec les octets de section correspondants.
mask: byte[]	Si un bit est mis à 0, la correspondance avec la valeur de la section est ignorée.
neg: byte []	Si un bit est mis à 1, la correspondance avec le bit de section est négative.

Une section correspond au filtre si tous ses bits masqués filtrés positivement concordent avec la valeur de filtre correspondante et si aucun bit masqué filtré négativement ne concorde avec la valeur de filtre correspondante (à condition qu'il y ait au moins un bit filtré négativement). Une correspondance de section (représentée par **data** pour les octets 1 et 3-18 de la section) est définie par la fonction **sectionFilterMatch**.

```
bool sectionFilterMatch(byte *data, *filter, *mask, *neg) {
    int i;
    bool posMatch, negMatch;

    posMatch = True;
    negMatch = True;

    /* si tous les octets neg sont à 0; le filtre négatif est toujours satisfait */
    for (i=0; i< DcrSectionFilterMaxlen; i++)
        negMatch &&= neg[i] == 0;

    /* mettre en correspondance les données de la section avec les critères de
filtrage positif et négatif*/
    for (i=0; i< DcrSectionFilterMaxlen; i++) {
        posMatch &&= (data[i] & mask[i] & ~neg[i]) == (filter[i] & mask[i] & ~neg[i]);
        negMatch ||= (data[i] & mask[i] & neg[i]) != (filter[i] & mask[i] & neg[i]);
    }
    return posMatch && negMatch;
}
```

9.6.2.3.5.3 Message reqDcrTsSectionAcqDefault

C→H setDcrTsSectionAcqDefault(ushort **mH,** uchar **filterNr,** dcrSectionFilterSpec **sectionFilter)**

- Ce message définit les filtres de section par défaut qui seront utilisés par l'**Hôte ECI** pour acquérir des informations à partir du flux pour le **Client ECI** après réception d'un message resDcrTsDescrStart. Cette fonction peut être utilisée par exemple par le **Client ECI** pour accélérer l'acquisition de sections d'ECM par l'**Hôte ECI** lors d'un changement de chaîne.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport sur lequel définir le filtre de section par défaut.
filterNr: uchar	Numéro du filtre à programmer. La valeur doit être comprise entre 0 et 7.
sectionFilter: dcrSectionFilterSpec	Spécification de filtre de section conformément au type dcrSectionFilterSpec défini au § 9.6.2.3.5.2.

Postcondition:

- Ce filtre de section sera appliqué par l'**Hôte ECI** immédiatement après la réception d'un message **resDcrTsDescrStart** ayant produit un résultat. Dans la mesure du possible, l'**Hôte ECI** doit anticiper une réponse **resDcrTsDescrStart** produisant un résultat.

9.6.2.3.5.4 Message reqDcrTsSectionAcq

C→H setDcrTsSectionAcq(ushort **mH,** uchar **filterNr,** dcrSectionFilterSpec **sectionFilter**

- Ce message définit les filtres de section qui seront utilisés par l'**Hôte ECI** pour acquérir des informations à partir du flux **mH** pour le **Client ECI**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport sur lequel définir le filtre de section par défaut.
filterNr: uchar	Numéro du filtre à programmer. La valeur doit être comprise entre 0 et 7.
sectionFilter: dcrSectionFilterSpec	Spécification de filtre de section conformément au type dcrSectionFilterSpec défini au § 9.6.2.3.5.2.

Sémantique détaillée:

- L'utilisation de ce message après avoir défini un filtre de section par défaut modifiera le filtre de section jusqu'à l'émission du prochain message resDcrTsDescrStart sur le même **Pointeur de média**, laquelle rétablira le filtre de section par défaut.

Postcondition:

- Ce filtre de section sera appliqué par l'**Hôte ECI**.

9.6.2.3.5.5 Message reqDcrTsSection

H→C reqDcrTsSection(ushort **mH,** uchar **filterNr,** uint **sectionDataLen**, byte **sectionData[]) →**
C→H resDcrTsSectionAcq (ushort **mH,** uchar **filterNr)**

- Ce message envoie au **Client ECI** une ou plusieurs sections acquises par l'**Hôte ECI** dans le contexte du flux de transport identifié par le paramètre **mH** et du filtre identifié par le paramètre **filterNr**.

Les codes d'erreur associés sont définis dans le Tableau 9.6.2.3.5.5-1.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport sur lequel définir le filtre de section par défaut.
filterNr: uchar	Numéro du filtre à programmer. La valeur doit être comprise entre 0 et 7.
sectionDataLen: uint	Nombre d'octets dans **sectionData**.
sectionData: byte []	Séquence de fonctions private_section (ordre des octets du réseau) telle que définie au § 2.4.4.11 de la norme [ISO/CEI 13818-1]. Toute section comportant une erreur de CRC n'est pas transmise au **Client ECI**.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.
filterNr: uchar	Numéro du filtre programmé.

Préconditions de la requête:

1) Les sections doivent avoir été acquises par l'**Hôte ECI** conformément à la spécification de filtre de section ou le délai d'attente pour le filtre a expiré.

2) La réception du message **reqDcrTsSection** précédent a été confirmée par une réponse **resDcrTsSection**.

Postconditions de la réponse:

1) Le message **reqDcrTsSection** suivant issu du même filtre peut être envoyé par l'**Hôte ECI**.

Tableau 9.6.2.3.5.5-1 – Codes d'erreur du message reqDcrTsSection

Nom	Description
ErrDcrTsSectionTimeout	Voir le Tableau 9.6.2.3.7-1.
ErrDcrTsSectionCrcErr	

9.6.2.3.5.6 Message reqDcrTsTable

C→H reqDcrTsTable(ushort **mH,** uchar **tableId,** uint **timeout,** uint **maxLen)**
H→C resDcrTsTable(ushort **mH,** uint **tableDataLen,** byte **tableData[])**

- Ce message demande à l'**Hôte ECI** d'envoyer les sections composant une table standard ou une sous-table en fonction du programme en cours de désembrouillage sur le **Pointeur de média mH**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport sur lequel définir le filtre de section par défaut.
tableId: uchar	Numéro du filtre à programmer. Les valeurs valides sont répertoriées dans le Tableau 9.6.2.3.5.6-1.
timeout: uint	Délai d'expiration en millisecondes. Si ce champ est défini sur 0, il n'y a pas de délai d'expiration.
maxLen: uint	Nombre maximal d'octets sectionData à renvoyer. L'**Hôte ECI** arrondira à l'entier inférieur du nombre le plus élevé de sections dans cette limite.

Tableau 9.6.2.3.5.6-1 – Valeurs de ca_pmt_list_management

Nom	Valeur	Description
DcrTsTableMpegPat	0x0000	Table d'association de programme (PAT) conformément à la norme [ISO/CEI 13818-1].
DcrTsTableMpegCat	0x0001	Table d'accès conditionnel (CAT) conformément à la norme [ISO/CEI 13818-1].
DcrTsTableMpegPmt	0x0002	Table PMT du programme sélectionné conformément à la norme [ISO/CEI 13818-1]. Le résultat est vide si une table PMT composite est utilisée par l'application.
DcrTsTableDvbNit	0x0140	Table NIT du réseau de fourniture réel comme spécifiée dans les documents [ETSI EN 300 468] et [ETSI TS 101 211]. Sur les réseaux câblés utilisant NIT_{other} pour transporter les tables associées aux régions des réseaux en question, la table NIT_{other} applicable à la région de l'**Équipement CPE** sera désignée.
DcrTsTableDvbSdt	0x0142	Table SDTactual_current comme spécifiée dans les documents [ETSI EN 300 468] et [ETSI TS 101 211].
DcrTsTableDvbBat	0x014A	Table BATactual comme spécifiée dans le document [ETSI EN 300 468] pour le bouquet activement utilisé par l'**Hôte ECI** et/ou son application.
DcrTsTableDvbEitPf	0x014E	Table EITactual actuelle et suivante comme spécifiée dans les documents [ETSI EN 300 468] et [ETSI TS 101 211].
DcrTsDescrStartUpdate	0x05	Même signification que **DcrTsDescrStartOnly**.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.
tableDataLen: uint	Nombre d'octets dans tableData.
tableData: byte []	Séquence de fonctions private_section (ordre des octets du réseau) représentant la (sous-)table, telle que définie au § 2.4.4.11 de la norme [ISO/CEI 13818-1].

Sémantique détaillée:

- L'**Hôte ECI** utilisera les filtres de section pour acquérir les dernières données pour toutes les tables pouvant être demandées par le **Client ECI** (ainsi que pour ses autres besoins). Les sections de table seront envoyées une fois par l'**Hôte ECI**, qui retardera la **Réponse** s'il n'a pas encore acquis la table demandée. La table sera "à jour" et utilisera les dernières données complètes à la disposition de l'**Hôte ECI**. Les codes d'erreur sont définis dans le Tableau 9.6.2.3.5.6-2.

NOTE:

– Une table pourra toujours être remplacée ultérieurement à n'importe quel moment par une version suivante dans un flux.

– Des taux de répétition minimaux pour la mise à jour des tables DVB SI pertinentes sont proposés dans le document [b-UIT-T J Suppl. 7].

– PAT, CAT et PMT: l'ancienneté des données est supérieure à 20 secondes.

Tableau 9.6.2.3.5.6-2 – Codes d'erreur du message reqDcrTsTable

Nom	Description
ErrDcrTsSectionTimeout	Voir le Tableau 9.6.2.3.7-1.
ErrDcrTsSectionCrcErr	

9.6.2.3.6 Contrôle de la source par le Client ECI

9.6.2.3.6.1 Généralités

Le **Client ECI** a la capacité de lire le type de source du flux de transport, de contrôler (rediriger) la source en question et de rediriger le programme et/ou les composantes décodés par l'**Hôte ECI**. Les messages sont répertoriés dans le Tableau 9.6.2.3.6.1-1.

Tableau 9.6.2.3.6.1-1 – Messages d'API relatifs au contrôle de la source par le Client ECI

Message	Type	Sens	Étiquette	Description
getDcrTsSource	get	C→H	0x18	Le **Client ECI** obtient la source du flux de transport.
reqDcrTsRelocate	A	C→H	0x19	Le **Client ECI** déplace la source du flux de transport.
reqDcrTsSelectPrg	A	C→H	0x1A	Le **Client ECI** sélectionne un programme dans le flux de transport en fonction du numéro de programme.
reqDcrTsSelectPmt	A	C→H	0x1B	Le **Client ECI** sélectionne un programme dans le flux de transport en fonction de la PMT.
reqDcrTsSelectCancel	A	C→H	0x1C	Le **Client ECI** annule sa sélection de programme précédente.

9.6.2.3.6.2 Message getDcrTsSource

C→H tsSourceType **getDcrTsSource(**ushort **mH**)

- Ce message renvoie le type de source du **Pointeur de média** en ce qui concerne le type de réseau et le localisateur dans le réseau.

Définition du paramètre:

mH: ushort	**Pointeur de média** du flux de transport visant à obtenir le type et l'emplacement du flux syntonisé.

Définition de la propriété:

Les définitions de la propriété sont fournies dans le Tableau 9.6.2.3.6.2-1.

Tableau 9.6.2.3.6.2-1 – Définition de type pour la structure tsSourceType

```
#define MaxTsSourceDescr 254

typedef struct tsSourceType{
    ushort tsSourceTag ;
    byte tsSourceDescr[MaxTsSourceDescr] ;
} tsSourceType ;
```

tsSourceTag: ushort	Type de la source du flux de transport. Les valeurs définies sont répertoriées ci-dessous, notamment la signification correspondante de **tsSourceDescr**.
tsSourceDescr: byte[MaxTsSourceDescr]	La signification dépend du champ tsSourceTag et est décrite dans le Tableau 9.6.2.3.6.2–2.

Tableau 9.6.2.3.6.2–2 – Signification de tsSourceTag

Nom	Valeur	Description
tsSourceDvbTuner	0x0001	La source du flux de transport est un syntoniseur DVB. Le champ tsSourceDescr contient un descripteur unique énoncé dans le Tableau 9.6.2.3.6.2-3 dans l'ordre des octets du réseau.
tsSourceDvbFile	0x0002	La source du flux de transport est un fichier ou un autre type de source non syntonisable tel qu'un réseau IP (voir [b-ETSI TS 102 034]). Le champ tsSourceDescr n'est pas défini.
tsDvbDuplet	0x8003	La source du flux de transport peut être trouvée à l'aide de l'identificateur du réseau d'origine et de l'identificateur du flux de transport dans le réseau actuel. Le champ tsSourceDescr contiendra l'ordre des octets du réseau pour `struct dvbDuplet {ushort onid; ushort tsid};` Cette valeur ne sera pas renvoyée par le message getDcrTsSource (qui renverra plutôt la valeur tsSourceDvbTuner) mais peut être utilisée dans un message reqDcrTsRelocate.
RFU	Autre	Réservé à une utilisation future.

Les valeurs supérieures à 0x7FFF ne sont pas des localisateurs absolus et ne seront pas renvoyées par le message **getDcrTsSource**.

Tableau 9.6.2.3.6.2-3 – Descripteurs d'une source syntoniseur DVB

Nom du descripteur de transmission DVB	Valeur de l'étiquette du descripteur DVB
terrestrial_delivery_system_descriptor	0x5A
T2_delivery_system_descriptor	0x7F, 0x04
satellite_delivery_system_descriptor	0x43
S2_delivery_system_descriptor	0x79
cable_delivery_system_descriptor	0x44
C2_delivery_system_descriptor	0x7F, 0x0D

Les descripteurs doivent être utilisés comme défini dans le document [ETSI EN 300 468] et contenir une seule fréquence de destination.

9.6.2.3.6.3 Message reqDcrTsRelocate

C→H reqDcrTsRelocate(ushort **mH**, tsSourceType **tsLoc**) →
H→C resDcrTsRelocate(ushort **mH)**

- Ce message demande à l'**Hôte ECI** de déplacer la source du flux de transport vers **tsLoc**. Les codes d'erreur associés sont définis dans le Tableau 9.6.2.3.6.3–1.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport à syntoniser à nouveau/à déplacer.
tsLoc: tsSourceType	Emplacement vers lequel déplacer le flux tel que défini dans le Tableau 9.6.2.3.6.2-1.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport déplacé.

Sémantique détaillée:

- Si une ressource d'accès au réseau (par exemple syntoniseur/démodulateur pour la radiodiffusion) autre que celle actuellement attribuée au **Pointeur de média** est requise, il est possible que l'**Hôte ECI** rejette la **Requête** en raison de contraintes de ressources.
- Lorsqu'une nouvelle syntonisation est réalisée avec succès, tout filtrage et/ou désembrouillage existant est terminé. L'acquisition par défaut commencera une fois le flux de transport acquis.

Tableau 9.6.2.3.6.3-1 – Codes d'erreur du message reqDcrTsRelocate

Nom	Description
ErrDcrTsNetworkAccessCapability	Voir le Tableau 9.6.2.3.7-1.
ErrDcrTsNetworkAccessResource	
ErrDcrTsNetworkAccessFail	

9.6.2.3.6.4 Message reqDcrTsSelectPrg

C→H reqDcrTsSelectPrg(ushort **mH,** ushort **prgNumber) →**
H→C resDcrTsSelectPrg(ushort **mH)**

- Ce message définit sur **prgNumber** la sélection du programme devant être désembrouillé par l'**Hôte ECI** dans le flux de transport actuel.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
prgNumber: ushort	Numéro de programme dans les tables MPEG PAT et PMT (voir la norme [ISO/CEI 13818-1]) du flux de transport définissant le service devant être sélectionné par l'**Hôte ECI**.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Sémantique détaillée:

- L'**Hôte ECI** localisera la PAT dans le flux de transport indiqué par le paramètre **mH**. Il localisera le PID de la PMT en mettant en correspondance **prgNumber** avec program_number. Il obtiendra la PMT à partir du PID localisé et utilisera les fonctions habituelles de l'**Hôte ECI** pour la sélection des composantes du programme à restituer. Si cette opération est réalisée avec succès, l'**Hôte ECI** émettra une **Requête reqDcrTsDescrStart demandant le démarrage** du désembrouillage du programme.

Postconditions de la requête:

1) Si l'**Hôte ECI** était en train de désembrouiller un programme non sélectionné par une **Requête reqDcrTsSelectPrg** ou **reqDcrTsSelectPmt**, il gardera en mémoire les paramètres de sélection du programme pour pouvoir y revenir plus tard lors d'une **Requête reqDcrTsSelectCancel**.

Postconditions de la réponse:

1) Si aucune erreur n'est renvoyée, l'**Hôte ECI** enverra par la suite une requête **reqDcrTsDescrStart**.

Les codes d'erreur relatifs à ce message API sont répertoriés dans le Tableau 9.6.2.3.6.4-1.

Tableau 9.6.2.3.6.4-1 – Codes d'erreur du message reqDcrTsSelectPrg

Nom	Description
ErrDcrTsPrgNumberNotInPsi	Voir le Tableau 9.6.2.3.7-1.
ErrDcrTsComponentSelectError	

9.6.2.3.6.5 Message reqDcrTsSelectPmt

C→H reqDcrTsSelectPmt(ushort **mH,** uint **pmtLen,** byte **pmt[]) → H→C resDcrTsSelectPmt(**ushort **mH)**

- Ce message sélectionne un nouveau programme devant être désembrouillé par l'**Hôte ECI** en envoyant une table MPEG PMT définissant les composantes du programme dans le flux de transport identifié par le paramètre **mH**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
pmtLen: uint	Nombre d'octets du paramètre **pmt**.
pmt: byte	private_section contenant une table PMT conformément à la norme [ISO/CEI 13818-1].

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Sémantique détaillée:

- Cette commande permet à un **Client ECI** de sélectionner dans un flux de transport des composantes n'ayant pas de tables PAT et PMT appropriées. L'**Hôte ECI** utilisera le paramètre **pmt** pour sélectionner les composantes du programme à restituer. Si cette opération est réalisée avec succès, l'**Hôte ECI** émettra une **Requête reqDcrTsDescrStart demandant le démarrage** du désembrouillage du programme.

Postconditions de la requête:

1) Si l'**Hôte ECI** était en train de désembrouiller un programme non sélectionné par une **Requête reqDcrTsSelectPrg** ou **reqDcrTsSelectPmt**, il gardera en mémoire les paramètres de sélection du programme pour pouvoir y revenir ultérieurement lors d'une **Requête reqDcrTsSelectCancel**.

Postconditions de la réponse:

1) Si aucune erreur n'est renvoyée, l'**Hôte ECI** enverra par la suite une requête **reqDcrTsDescrStart**.

Les codes d'erreur relatifs à ce message API sont répertoriés dans le Tableau 9.6.2.3.6.5-1.

Tableau 9.6.2.3.6.5-1 – Codes d'erreur du message reqDcrTsSelectPmt

Nom	Description
ErrDcrTsComponentSelectError	Voir le Tableau 9.6.2.3.7-1.

9.6.2.3.6.6 Message reqDcrTsSelectCancel

C→H reqDcrTsSelectCancel(ushort **mH) →**
H→C resDcrTsSelectCancel(ushort **mH)**

- Ce message annule une **Requête reqDcrTsSelectPrg** ou **reqDcrTsSelectPmt** du **Client ECI** précédente et ramène au programme sélectionné initialement par l'**Hôte ECI** dans le flux de transport identifié par le paramètre **mH**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Postconditions de la réponse:

1) Il est possible que l'**Hôte ECI** envoie par la suite un message **reqDcrTsDescrStart** pour reprendre le désembrouillage du programme initial.

9.6.2.3.7 Codes d'erreur de l'API de session de média pour les médias TS

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.6.2.3.7-1.

Toutes les requêtes de **Pointeur de média** spécifiques à un flux de transport renvoient un code d'erreur pour le paramètre du **Pointeur de média** si elles sont appliquées à un **Pointeur de média** non associé à un flux de transport.

Tableau 9.6.2.3.7-1 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrDcrTsUserDelay	-256	Le système a attendu longtemps une action de la part de l'**Utilisateur** nécessaire pour réaliser l'opération. L'opération n'a pas abouti.
ErrDcrTsCardMissing	-257	La **Carte à puce** requise pour la session n'est pas accessible/disponible.
ErrDcrTsServiceMissing	-258	Un service externe à l'**Équipement CPE** requis pour aider le **Client ECI** dans les opérations de déchiffrement n'est pas disponible.
ErrDcrTsResourceMissing	-259	Une ressource non définie interne à l'**Équipement CPE** nécessaire pour accéder au contenu ou le déchiffrer n'est pas disponible.
ErrDcrTsMmiMissing	-260	L'accès du **Client ECI** à l'interface homme-machine n'est pas disponible.
ErrDcrDescrContinue	-261	L'**Hôte ECI** continue à tenter de désembrouiller le contenu de ce flux de transport.
ErrDcrTsSectionTimeout	-262	Le délai d'attente pour l'acquisition d'une section a expiré.
ErrDcrTsSectionCrcErr	-263	Des sections ont été extraites avant l'expiration du délai d'attente, mais avec des erreurs de CRC. Cela signifie généralement que le flux est fortement corrompu.
ErrDcrTsNetworkAccessCapability	-264	Il manque à l'**Hôte ECI** une ressource d'accès au réseau pour localiser le flux de transport requis.
ErrDcrTsNetworkAccessResource	-265	L'**Hôte ECI** ne peut acquérir la ressource d'accès au réseau nécessaire pour accéder au flux de transport demandé.
ErrDcrTsNetworkAccessFail	-266	La ressource d'accès au réseau n'a pas réussi à acquérir (de manière fiable) le flux de transport demandé.
ErrDcrTsPrgNumberNotInPsi	-267	Aucune PMT comportant un numéro de programme correspondant n'a été localisée dans la PAT.
ErrDcrTsComponentSelectError	-268	Une composante de la PMT n'a pas pu être sélectionnée pour démultiplexage/désembrouillage.
ErrDcrTsPidNotDescrambled	-269	Aucun PID n'a été sélectionné par l'**Hôte ECI** pour un désembrouillage.
ErrDcrTsCwIdNotValid	-270	Un identificateur de mot de contrôle non valide a été référencé.
RFU	autre	Réservé à une utilisation future.

9.6.2.4 Déchiffrer du contenu basé sur fichier et sur flux

9.6.2.4.1 Introduction

Ce paragraphe définit une API **Client ECI/Hôte ECI** permettant à l'**Équipement CPE** et aux applications téléchargées d'interagir avec un **Client ECI** de sécurité par l'intermédiaire de l'**Hôte ECI** afin de déchiffrer du contenu au format ISOBMFF [ISO/CEI 23001-9] ou tout autre fichier ou flux pour lesquels l'**Hôte ECI** (ou l'**Équipement CPE** sous-jacent ou l'application téléchargée agissant par son biais):

- peut extraire les données de contrôle de sécurité requises du fichier ou du flux et les transmettre au **Client ECI**;
- permet l'application correcte (synchronisation) des clés de désembrouillage générées par le **Client ECI** au contenu, au moyen d'identificateurs de clé.

Les fichiers ISOBMFF ([ISO/CEI 23001-9]) représentent un format de paquet de fichiers commun à de nombreuses méthodes de téléchargement adaptatives et en différé. Il existe également une méthode de chiffrement commune définie pour ces formats de fichier: CENC [ISO/CEI 23001-7]. Par ailleurs, la norme de format de streaming adaptatif MPEG-DASH ([ISO/CEI 23009-1] et [ETSI TS 103 285]) est basée sur le format ISOBMFF et différents systèmes de DRM (parfois hérités) utilisent leur propre sous-format ISOBMFF propriétaire (avec identificateur de "marque" spécifique).

Une section de l'API permet au **Client ECI** d'indiquer de quelles données il a besoin dans le fichier ISOBMFF pour effectuer le décodage, permettant ainsi aux applications de l'**Équipement CPE** d'utiliser les applications DRM propriétaires au format ISOBMFF (non conformes CENC). Les détails du désembrouillage de l'échantillon devraient être gérées par l'**Hôte ECI**, à savoir être conformes CENC ou nécessiter des extensions propriétaires dans l'**Hôte ECI**.

L'API comporte les sections suivantes:

1) Démarrage et arrêt du désembrouillage;
2) Définition des filtres d'acquisition des données de sécurité spécifiques au **Client ECI**;
3) API relative à la clé de déchiffrement (mot de contrôle).

9.6.2.4.2 Spécifications applicables

Les fichiers ISOBMFF tels que mentionnés dans ce paragraphe doivent être conformes à la norme [ETSI TS 103 285]. Les fichiers ISOBMFF conformes CENC (comme l'exige le déchiffrement standard) doivent être conformes à la norme [ISO/CEI 23001-7].

Les données de streaming conformes DASH doivent être conformes à la norme [ISO/CEI 23009-1]. Les **Hôtes ECI** appliquant la norme DASH doivent (au minimum) être conformes aux normes [ISO/CEI 23001-7], [ISO/CEI 23001-9] et [ETSI TS 103 285], dans la mesure applicable au fonctionnement de l'**Équipement CPE**.

9.6.2.4.3 Exigences de traitement de l'Hôte ECI

9.6.2.4.3.1 Détection de l'identification du système de déchiffrement

L'**Hôte ECI** sera capable d'acquérir la liste des systèmes de déchiffrement applicables à partir du conteneur de contenu en se basant sur les règles suivantes:

1) Pour tous les fichiers ISOBMFF et MP4, l'**Hôte ECI** acquerra la boîte de type de fichier (ftyp) et la boîte de type de segment (styp) et utilisera les champs `major_brand` et `compatible_brands[]` pour mettre en correspondance le contenu avec les **Clients ECI**.
2) Pour les fichiers encodés ISOBMFF CENC, l'**Hôte ECI** récupérera les boîtes "pssh" (Protection System Specific Header) à partir de tous les emplacements possibles (voir la norme [ISO/CEI 23001-7]) et extraira du champ `SystemID` les UUID des systèmes DRM adaptés pour déchiffrer le contenu. Ces fichiers peuvent être reconnus par une boîte "sinf"

(Protection Scheme Information) contenant une boîte "schm" (Scheme Type Box) dont le champ `scheme_type` est mis à 'cenc' ou 'cbc1' et la version majeure du champ `scheme_version` est mise à 0x0001. La définition et l'emplacement des boîtes "sinf" sont spécifiés dans la norme [ISO/CEI 23001-7].

3) Pour le contenu MPEG-DASH, l'**Hôte ECI** acquerra tous les descripteurs ContentProtection de la MPD (Description de la présentation des médias) contenant un UUID spécifique (c'est-à-dire commençant par "urn:uuid:xxxxx", xxxxx correspondant à l'UUID) pour l'attribut @SchemeIdUri afin d'effectuer la mise en correspondance avec les UUID DRM du **Client ECI**, ou contenant un identificateur de système d'accès conditionnel conformément à la norme [ETSI TS 103 285] dans l'attribut @value (voir [b-DASH-IF ID] pour la définition de cet identificateur générique). Cette opération vise la mise en correspondance avec les capacités du **Client ECI**. L'**Hôte ECI** convertira toutes les boîtes "pssh" incluses dans la représentation binaire ISOBMFF correspondante.

Le processus de mise en correspondance du contenu avec les **Clients ECI** est décrit au § 9.6.2.4.5.2.1.

9.6.2.4.3.2 Détection du type d'embrouillage

L'**Hôte ECI** signalera au **Client ECI** le mode de désembrouillage applicable en se basant sur les règles suivantes:

1) Concernant les fichiers encodés ISOBMFF CENC, il sera capable d'appliquer les règles définies dans la norme [ISO/CEI 23001-7] pour la détection du mode de chiffrement (AES-CTR ou AES-CBC), y compris la sélection de l'octet clair/embrouillé, le bourrage, ainsi que l'extraction et l'application du vecteur d'initialisation comme défini dans la norme [ISO/CEI 23001-7].

2) Concernant le contenu MPEG-DASH au format ISOBMFF, le mode AES-CTR (avec rotation des clés) sera appliqué pour le désembrouillage, comme défini dans la norme [ETSI TS 103 285].

9.6.2.4.3.3 Filtrage des données de sécurité du conteneur de contenu par défaut

L'**Hôte ECI** transmettra les boîtes contenant des informations (opaques) dans le conteneur désigné pour le **Client ECI** au moment où cela sera nécessaire pour le processus de désembrouillage. Cela vaut en particulier pour les boîtes suivantes dans les fichiers encodés ISOBMFF CENC et pour le contenu DASH au format ISOBMFF:

1) Pour:
   a) Les boîtes "pssh" des boîtes "moov" et "moof" correspondant à l'UUID de l'identificateur de système DRM du **Client ECI**, pertinentes pour le contenu en cours de décodage ou devant être décodé dans un avenir proche.
   b) Les boîtes "sinf", au cas où le **Client ECI** aurait besoin d'y accéder.

9.6.2.4.3.4 Désembrouillage du contenu

L'**Hôte ECI** sera chargé de l'interprétation du mode d'embrouillage, de l'identification des données à désembrouiller et de leur traitement à l'aide du désembrouilleur en utilisant les identificateurs de clé appropriés pour identifier les clés mises à disposition par le **Client ECI**.

Afin que le **Client ECI** puisse calculer les clés associées, l'**Hôte ECI** lui transmettra les données de contrôle de sécurité requises depuis le conteneur de contenu, en temps utile.

9.6.2.4.4 API de session de média pour les médias basés sur fichier et sur flux

9.6.2.4.4.1 Généralités

L'**Hôte ECI** peut lancer le déchiffrement du contenu sur un **Pointeur de média** ouvert en utilisant les ressources du **Client ECI** réservées. L'**Hôte ECI** fournira les données d'initialisation pour que le **Client ECI** puisse commencer à évaluer les droits d'accès.

Tableau 9.6.2.4.4.1-1 – API de déchiffrement pour le contenu du flux de transport du Pointeur de média

Message	Type	Sens	Étiquette	Description
reqDcrFileStart	A	H→C	0x01	Demande au **Client ECI** de désembrouiller un fichier ou un flux ou de renvoyer son statut de désembrouillage.
reqDcrFileStop	A	H→C	0x02	L'**Hôte ECI** demande au **Client ECI** d'arrêter le traitement des clés pour l'opération de désembrouillage au niveau d'un **Pointeur de média**.
reqDcrFileQuit	A	C→H	0x03	Le **Client ECI** annule une opération de désembrouillage avec l'**Hôte ECI**.

9.6.2.4.4.2 Message reqDcrFileStart

H→C reqDcrFileStart(ushort **mH,** uchar **reqType,** uchar **dataType,** uint **initDataLen,** byte **initData[]) →**
C→H resDcrFileStart(ushort **mH,** uchar **dcrStat)**

- Ce message demande au **Client ECI** de renvoyer le statut de désembrouillage du contenu associé au **Pointeur de média** mH et/ou de démarrer une session de désembrouillage du contenu en question. L'**Hôte ECI** fournit les données initiales pour que le **Client ECI** commence l'acquisition et l'évaluation de la licence conformément au format du conteneur/de chiffrement.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du fichier.
reqType: uchar	Type de **Requête** (démarrage du désembrouillage ou demande concernant la licence) tel que défini dans le Tableau 9.6.2.4.4.2-1.
dataType: uchar	Type d'InitData.
initDataLen: uint	Longueur en octets du conteneur InitData.
initData: byte	Données d'initialisation issues du contenu telles que définies par dataType. Le codage d'initDat est défini dans le Tableau 9.6.2.4.2–2.

Tableau 9.6.2.4.4.2-1 – codage de reqType

Nom	Valeur	Description
ReqTypeDcr	0x01	Démarrage du désembrouillage; lancement du dialogue avec l'**Utilisateur** si nécessaire.
ReqTypeInq	0x02	Demande concernant les options de désembrouillage.
RFU	Autre	Réservé à une utilisation future.

Tableau 9.6.2.4.4.2-2 – codage d'initData

dataType	Valeur	Description
FmtIsoCenc	0x04	Boîtes "pssh" des fichiers ISOBMFF (voir la norme [ISO/CEI 23001-7]) rencontrées correspondant à l'identificateur DRM contenu dans la structure MatchSpecifier du **Client ECI**.
FmtIsoCencDash	0x05	Boîtes "pssh" des fichiers ISOBMFF (voir la norme [ISO/CEI 23001-7]) figurant dans la MPD (voir la norme [ISO/CEI 23007-1]) ou segment d'initialisation (voir la norme [ISO/CEI 23009-1]) rencontré correspondant à l'identificateur DRM contenu dans la structure MatchSpecifier du **Client ECI**.
FmtIsoProp	0x06	L'**Hôte ECI** peut transmettre des données au **Client ECI** sur la base des connaissances relatives aux formats propriétaires. Le **Client ECI** sera capable d'interpréter ces données sur la base de ces mêmes connaissances.
FmtIsoPropDash	0x07	FmtIsoProp incluant l'indication selon laquelle les données sont une source DASH.
RFU	Autre	Réservé à une utilisation future.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.
dcrStat: uchar	Statut du désembrouillage; voir le Tableau 9.6.2.4.4.2-3.

Tableau 9.6.2.4.4.2-3 – Statut du désembrouillage

Nom	Valeur	Description
DcrStatNo	0x00	Aucun désembrouillage possible (le système DRM est capable d'effectuer le désembrouillage).
DcrStatOk	0x01	Démarrage du désembrouillage; lancement du dialogue avec l'**Utilisateur** si nécessaire.
DcrStatDialog	0x02	Dialogue requis avec l'**Utilisateur**.
DcrStatPay	0x03	Paiement requis, éventuellement dialogue avec l'**Utilisateur** également.
DcrStatDrmNok	0xFE	Le système DRM n'est pas capable de désembrouiller le contenu.
RFU	Autre	Réservé à une utilisation future.

Sémantique détaillée:

- Lors des demandes, aucun dialogue avec l'**Utilisateur** ne sera lancé par le **Client ECI** mais ce dernier évaluera la capacité à désembrouiller le contenu en vérifiant les conditions de licence avec le serveur de licences, sans dialogue avec l'**Utilisateur**.

Préconditions de la requête:

1) Pointeur de média en attente.

Préconditions de la réponse:

1) Si le **Client ECI** est capable de désembrouiller le contenu et que reqType est OK, le **Client ECI** sera prêt à générer les clés de désembrouillage.

Les codes d'erreur relatifs au message de requête de démarrage du déchiffrement sont répertoriés dans le Tableau 9.6.2.4.4.2-4.

Tableau 9.6.2.4.4.2-4 – Codes d'erreur du message reqDcrFileStart

Nom	Description
ErrDcrFileUserDelay	Voir le Tableau 9.6.2.4.7-1.
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	

9.6.2.4.4.3 Message reqDcrFileStop

H→C reqDcrFile Stop(ushort **mH) →**
C→H resDcrFile Stop(ushort **mH)**

- Ce message permet à l'**Hôte ECI** d'arrêter l'opération de déchiffrement du fichier.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du fichier.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du fichier.

Préconditions de la réponse:

1) Le **Client ECI** a terminé toute opération liée au déchiffrement du contenu.

9.6.2.4.4.4 Message reqDcrFileQuit

C→H reqDcrFileQuit(ushort **mH,** uint **reason) →**
H→C resDcrFileQuit(ushort **mH)**

- Ce message permet au **Client ECI** d'informer l'**Hôte ECI** qu'il a mis fin au traitement des clés lors d'une opération de déchiffrement d'un fichier. Les codes d'erreur associés sont définis dans le Tableau 9.6.2.4.4.4–1.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
reason: uint	Les valeurs sont définies dans le Tableau 9.7.2.5.9–1.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du fichier.

Préconditions de la réponse:

1) Toutes les activités de l'**Hôte ECI** liées au désembrouillage du **Pointeur de média mH** se sont terminées ou une erreur a été renvoyée.

Postconditions de la réponse:

1) Toute activité du **Client ECI** liée au **Pointeur de média mH** prend fin immédiatement ou une erreur est renvoyée.

Tableau 9.6.2.4.4.4-1 – Codes d'erreur du message reqDcrFileQuit

Nom	Description
ErrDcrFileDescrContinue	Voir le Tableau 9.6.2.4.7-1.

9.6.2.4.5 Acquisition des données de sécurité spécifiques au Client ECI

9.6.2.4.5.1 Généralités

L'**Hôte ECI** effectuera une acquisition de données standard sur les données devant être décodées afin d'obtenir les informations requises par le **Client ECI** pour calculer les clés. Le **Client ECI** peut indiquer des données spécifiques à acquérir en plus des données standard fournies par l'**Hôte ECI**. Ce dernier tiendra à jour un nombre limité de filtres pour l'acquisition de ces données.

Tableau 9.6.2.4.5.1-1 – API de filtrage des données

reqDcrFileFilter	req	C→H	0x04	Le **Client ECI** demande à l'**Hôte ECI** de définir un filtre de données pour l'acquisition de données de sécurité.
reqDcrFileData	A	C→H	0x05	Le **Client ECI** demande à l'**Hôte ECI** d'acquérir des données par l'intermédiaire du filtre de fichiers.

9.6.2.4.5.2 Spécification du filtre de fichiers

9.6.2.4.5.2.1 Définition du filtre de fichiers générique

La spécification du filtre de données de fichier repose sur une spécification sous-jacente du format de fichier. Un filtre est défini dans le contexte d'un format de fichier défini. La spécification du filtre de fichiers générique est définie dans le Tableau 9.6.2.4.5.2.1-1.

Tableau 9.6.2.4.5.2.1-1 – Spécification du filtre de fichiers générique

```
typedef struct dcrFileFilterSpec {
   ushort filterType;         // défini dans le Tableau 9.6.2.4.5.2.1-3
   ushort filterLen;
   byte   filter[filterLen]; // format reposant sur filterType
} dcrFileFilterSpec;
```

Tableau 9.6.2.4.5.2.1-2 – Types de filtres de fichiers

FileFilterIsobmff	0x0001	Filtre de fichiers pour les données au format ISOBMFF comme défini au § 9.6.2.4.5.2.2.
RFU	Autre	Réservé à une utilisation future.

9.6.2.4.5.2.2 Définition du filtre de fichiers spécifique au format ISOBMFF

La spécification du filtre pour les fichiers au format ISOBMFF est définie dans le Tableau 9.6.2.4.5.2.2-1.

Tableau 9.6.2.4.5.2.2-1 – Spécification du filtre de fichiers ISOBMFF

```
#define MaxFilterFile 16 // nombre maximal d'octets de la boîte filtrés
#define MaxContainers 4  // nombre maximal de boîtes de ce type de conteneur pour
une boîte
#define MaxUuidLen 16// longueur en octets d'un UUID

typedef struct BoxSpec {
   uint   boxType                  // code FourCC du type de boîte
   byte   extendedType[MaxUuidLen]
                                   // UUID lorsque boxType=='uuid', sinon aucune
signification
   byte   filter[MaxFileFilter];       // correspond aux octets de la boîte
suivante
   byte   filterMask[MaxFilter];
   ushort dataLen;             // volume maximal de données de boîte à acquérir
} BoxSpec;

typedef struct dcrFileFilterIsobmff {
   BoxSpec    container[MaxContainer];
   BoxSpec    box;
} dcrFileFilterIsobmff;

bool function boxMatch
  (byte *boxData, byte *filter, byte*filterMask; int boxLen) {
{
   bool match = true;
   int i;
```

```
    for( i=0; i<MaxFilterFile && i<boxLen && match; i++) {
        match &&= (boxData[i] & filterMask[i] == filter & filterMask[i]) ;
    }
    return match;
}
```

L'**Hôte ECI** analysera le fichier et acquerra les boîtes correspondant au champ **box** qui sont contenues dans des boîtes correspondant à celles du tableau **container**. L'**Hôte ECI** n'analysera pas les boîtes non définies dans les normes [ISO/CEI 14496-12] ou [ISO/CEI 23001-7].

Dans le champ **container** de la structure **dcrFileFilterIsobmff**, **boxType** peut être défini sur '****' pour indiquer un caractère générique. Dans ce cas, les autres champs de **container** n'auront aucune signification et seront mis à 0 pour indiquer l'absence de correspondance.

Les champs **filter** et **filterMask** de la structure **BoxSpec** seront mis en correspondance avec les premiers octets suivant le champ de type d'une boîte devant être traitée. Pour les "boîtes complètes" (voir la norme [ISO/CEI 14496-12]), cela correspond aux champs de version et d'indicateur. La correspondance se fera sur la base de la fonction boxMatch, le paramètre **boxLen** étant défini sur le nombre d'octets suivant les champs boxtype et extended_type de la boîte, le paramètre **boxData** sur le début de ces octets, le paramètre **filter** sur le champ **boxSpec.filter** et le paramètre **filterMask** sur le champ **boxSpec.filterMask**.

Les données renvoyées par le filtre sont les boîtes (dans l'ordre) correspondant au filtre à mesure que le fichier est analysé par l'**Hôte ECI**. Ce dernier peut regrouper les boîtes selon ses besoins, mais ne doit pas retarder la transmission des boîtes au **Client ECI** inutilement, sans quoi celui-ci pourrait se trouver dans l'incapacité de générer les clés de désembrouillage requises.

9.6.2.4.5.2.3 Message reqDcrFileFilter

C→H setDrcFileFilter(ushort **mH,** uchar **filterNr,** dcrFilleFilterSpec ***dataFilter)**

- Ce message demande à l'**Hôte ECI** de définir un filtre de données sur la base du paramètre dataFilter en vue de l'acquisition des données de sécurité pour le **Client ECI**.

Définition des paramètres:

mH: ushort	**Pointeur de média** du flux de transport.
filterNr: uchar	Numéro du filtre de fichiers sur l'**Hôte ECI**.
dataFilter: dcrFileFilterSpec *	Spécification de filtre pour l'extraction des données.

Postconditions de la requête:

- Ce filtre de section sera appliqué par l'**Hôte ECI** jusqu'à l'application d'une **Requête reqDcrFileStop** ou **reqDcrFileQuit** ou jusqu'à ce qu'une **Requête** reqDcrFileFilter soit définie avec **dataFilter == NULL.**

9.6.2.4.5.2.4 Message reqDcrFileAcqData

H→C reqDcrFileAcqData(ushort **mH,** uchar **filterNr,** uint **dataLen**, byte **data[]) → C→H resDcrFileAcqData (**ushort **mH,** uchar **filterNr)**

Ce message demande à l'**Hôte ECI** d'acquérir et d'envoyer au **Client ECI** une ou plusieurs sections dans le contexte du flux ou fichier média identifié par le paramètre **mH** et du filtre identifié par le paramètre **filterNr**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du fichier sur lequel définir le filtre de section par défaut.
filterNr: uchar	Numéro du filtre à programmer. La valeur doit être comprise entre 0 et 7.
dataLen: uint	Nombre d'octets dans le paramètre **data**.
data[]: byte	Séquences de fonctions private_section (ordre des octets du réseau) telles que définies au § 2.4.4.11 de la norme [ISO/CEI 13818-1]. Toute section comportant une erreur de CRC n'est pas transmise au **Client ECI**.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux ou fichier média.
filterNr: uchar	Numéro du filtre programmé.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.6.2.4.5.2.4-1.

Tableau 9.6.2.4.5.2.4-1 – Codes d'erreur du message reqDcrFileAcqData

Nom	Description
ErrDcrAcqDataTimeout	Voir le Tableau 9.6.2.4.7-1.
ErrDcrAcqDataDataErr	

9.6.2.4.6 API relative au mot de contrôle destiné au désembrouillage d'un fichier

9.6.2.4.6.1 Généralités

L'API de désembrouillage du contenu permet de mettre une clé à disposition en vue du désembrouillage par le **Client ECI**. L'**Hôte ECI** doit d'abord initier la disponibilité d'un mot de contrôle en transmettant l'identificateur de clé au **Client ECI**. Une fois la clé disponible, l'**Hôte ECI** peut appliquer le mot de contrôle calculé au contenu (chiffré). Les messages fournis par l'API de désembrouillage du contenu d'un fichier sur un **Pointeur de média** sont répertoriés dans le Tableau 9.6.2.4.6.1-1.

Tableau 9.6.2.4.6.1-1 – API de désembrouillage du contenu d'un fichier sur un Pointeur de média

Message	Type	Sens	Étiquette	Description
reqDcrFileKeyComp	A	H→C	0x20	Initier toute activité de calcul ou autre de la part du **Client ECI** en vue de mettre à disposition un mot de contrôle pourvu d'un identificateur de clé.

9.6.2.4.6.2 Exigences de traitement de l'Hôte ECI

9.6.2.4.6.2.1 Contenu au format ISOBMFF CENC

Ce paragraphe définit les exigences de traitement de l'**Hôte ECI** pour le désembrouillage du contenu au format ISOBMFF CENC.

L'**Hôte ECI** doit transmettre en temps utile au **Client ECI** les informations relatives à l'identificateur de clé afin que le **Client ECI** puisse dériver/acquérir le mot de contrôle requis à temps. Parmi les contraintes associées, cette transmission doit se faire au moins 30 secondes avant l'utilisation anticipée du mot de contrôle.

Les informations relatives à l'identificateur de clé sont contenues dans plusieurs boîtes associées aux échantillons de média (séquences de données de média [en partie] chiffrées): voir par exemple [b-DASH-IF V3], § 5.4. Les données de ces boîtes permettent d'extraire les identificateurs de clé et les vecteurs d'initialisation, et d'identifier les données en clair et chiffrées des échantillons de média.

9.6.2.4.6.2.2 Contenu au format MPEG-DASH

Les spécifications **ECI** n'abordent actuellement pas les formats MPEG-DASH devant être pris en charge par l'**Hôte ECI**.

9.6.2.4.6.3 Message reqDcrFileKeyComp

H→C reqDcrFileKeyComp(ushort **mh,** byte **keyId**[MaxUuidLen]**) →**
C→H resDcrFileKeyComp(ushort **mH)**

- Ce message lance le calcul et toute autre activité requise par le **Client ECI** pour calculer un mot de contrôle identifié par un identificateur de clé et le mettre à disposition pour déchiffrer le contenu.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du flux de transport.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du flux de transport.

Préconditions de la réponse:

1) La clé est disponible, une erreur est survenue ou le délai d'attente a expiré.

Sémantique détaillée:

- Le **Client ECI** signalera une erreur si le mot de contrôle demandé ne peut être mis à disposition à temps (60 secondes). Les **Clients ECI** peuvent continuer à tenter d'acquérir la clé demandée même lorsqu'une erreur est signalée.
- Lorsqu'une erreur est signalée, l'**Hôte ECI** peut émettre à nouveau la **Requête**, dans une limite de 10 **Requêtes**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.6.2.4.6.3-1.

Tableau 9.6.2.4.6.3-1 – Codes d'erreur du message reqDcrFileKeyComp

Nom	Description
ErrDcrFileUserDelay	Voir le Tableau 9.6.2.4.7-1.
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	
ErrDcrFileKeyIdUnknown	
ErrDcrFileKeyOverflow	

9.6.2.4.7 Codes d'erreur de l'API de déchiffrement de contenu basé sur fichier et sur flux

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.6.2.4.7-1.

Toutes les requêtes de **Pointeur de média** spécifiques à un fichier renvoient un code d'erreur pour le paramètre du **Pointeur de média** si elles sont appliquées à un **Pointeur de média** non associé à un fichier.

Tableau 9.6.2.4.7-1 – Codes d'erreur des API de session de média pour les médias basés sur fichier et sur flux

Nom	Valeur	Description
ErrDcrFileUserDelay	-256	Le système a attendu longtemps une action de la part de l'**Utilisateur** nécessaire pour réaliser l'opération. L'opération n'a pas abouti.
ErrDcrFileCardMissing	-257	La **Carte à puce** requise pour la session n'est pas accessible/disponible.
ErrDcrFileServiceMissing	-258	Un service externe à l'**Équipement CPE** (par exemple un serveur DRM) requis pour aider le **Client ECI** dans les opérations de déchiffrement n'est pas disponible.
ErrDcrFileResourceMissing	-259	Une ressource non définie interne à l'**Équipement CPE** nécessaire pour accéder au contenu ou le déchiffrer n'est pas disponible.
ErrDcrFileMmiMissing	-260	L'accès du **Client ECI** à l'interface homme-machine n'est pas disponible.
ErrDcrFileDescrContinue	-261	L'**Hôte ECI** continue à tenter de désembrouiller le contenu de ce fichier.
ErrDcrAcqDataTimeout	-262	Le délai d'attente pour l'acquisition d'une donnée a expiré.
ErrDcrAcqDataDataErr	-263	Des sections ont été extraites avant l'expiration du délai d'attente, mais avec des erreurs. Cela signifie généralement que le fichier est corrompu ou n'est pas conforme aux spécifications applicables.
ErrDcrFileKeyIdUnknown	-300	Identificateur de clé inconnu du **Client ECI**/système de sécurité pour ce contenu.
ErrDcrFileKeyOverflow	-301	Trop de demandes liées à l'identificateur de clé ont été soumises en un court laps de temps; attendre les **Réponses** du **Client ECI** aux **Requêtes** précédentes en cours de traitement.
ErrDcrFileKeyWithdrawn	-302	La clé n'est plus disponible; les droits ont été retirés par le **Client ECI**.

9.7 API relatives à l'accès aux ressources de rechiffrement de l'Hôte ECI

9.7.1 Introduction aux API de rechiffrement

9.7.1.1 Liste des API définies au § 9.7

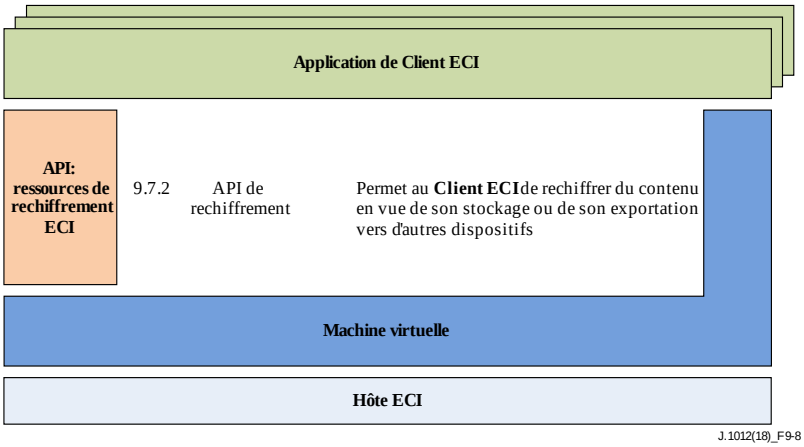


Figure 9.7.1-1 – Représentation des API définies au § 9.7

Le Tableau 9.7.1-1 répertorie les API présentées dans le § 9.7 et la Figure 9.7.1-1 illustre leur positionnement dans l'**Architecture ECI**. Voir également [b-Menezes].

Tableau 9.7.1-1 – liste des API définies au § 9.7

Paragraphe	Nom de l'API	Description
9.7.2.3	API de **Connexion d'exportation**	Permet au **Client ECI** d'établir une **Connexion d'exportation** pour le contenu importé.
9.7.2.5	API de **Connexion d'importation**	Permet au **Client ECI** d'importer du contenu envoyé chiffré via un réseau d'accès et déchiffré sous le contrôle d'un **Client ECI**.
9.7.2.6	API de redéchiffrement par le **Micro client**	Permet au **Client ECI** de déchiffrer du contenu importé et rechiffré.

9.7.1.2 Concept général du rechiffrement

Le rechiffrement dans l'interface **ECI** permet à un **Système Micro DRM** indépendant de protéger le contenu envoyé par un **Client ECI** CA ou DRM pour une utilisation à l'intérieur ou à l'extérieur de l'**Équipement CPE**. Le système de rechiffrement d'une implémentation conforme **ECI** est appelé **Système Micro DRM**. Il peut être utilisé par exemple pour le visionnement différé, les enregistreurs vidéo personnels et le streaming. Le **Client ECI** de rechiffrement est appelé **Micro Serveur**. Le client – qu'il soit conforme **ECI** ou non – qui est capable de déchiffrer et de rechiffrer le contenu est appelé **Micro client**. L'image du Client et les justificatifs d'identité utilisés pour le rechiffrement peuvent être téléchargés comme un **Client ECI** normal, auquel les ressources sont allouées par un serveur maître Micro DRM. La Figure 9.7.1.2-1 illustre l'ensemble du système (à l'exception du serveur maître Micro DRM). En cas de stockage local, le **Micro serveur** et le **Micro client** sont implémentés sur un seul et même dispositif.

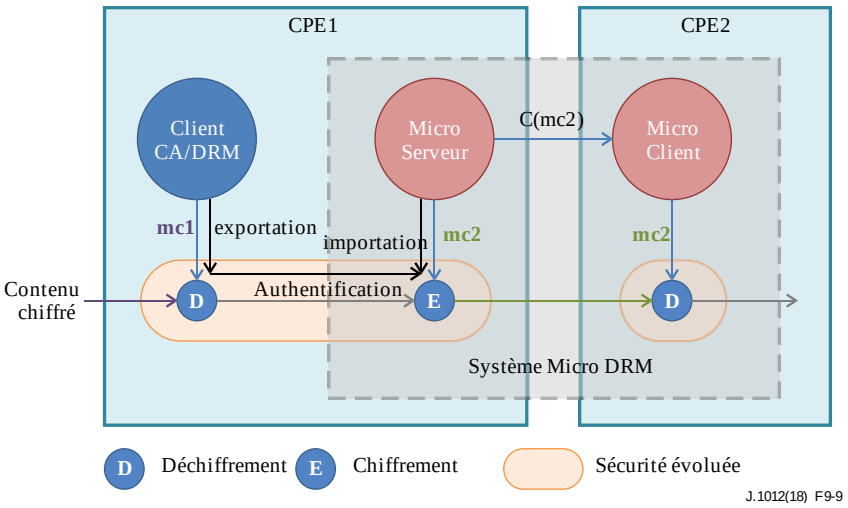


Figure 9.7.1.2-1 – Représentation graphique du Système Micro DRM

Le **Client ECI** CA/DRM déchiffrant le contenu initialement peut contrôler si l'exportation du contenu sur les **Systèmes Micro DRM** installés est autorisée ou non. A cet effet, il authentifie le **Micro serveur** par l'intermédiaire du **Système de sécurité évoluée**; l'authentification se fait sous le contrôle de l'opérateur CA/DRM. Une fois le contenu exporté, la responsabilité de sa protection revient au **Système Micro DRM**. Le déchiffrement, le rechiffrement et l'authentification en vue de l'exportation sont pris en charge de manière sécurisée par le **Système de sécurité évoluée**. Les principes sont illustrés à la Figure 9.7.1.2-1.

9.7.1.3 Vue d'ensemble de la structure des API de rechiffrement

La Figure 9.7.1.3-1 illustre dans un graphique plus détaillé le rôle des différentes API impliquées dans le rechiffrement. L'**Hôte ECI** fournit au **Client ECI** effectuant le décodage toutes les informations nécessaires par l'intermédiaire de l'API de déchiffrement. Le **Client ECI** effectuant le décodage établit le mot de contrôle de manière sécurisée en vue du déchiffrement du contenu par l'intermédiaire de l'API de sécurité évoluée. Les propriétés essentielles du contenu (marques) sont authentifiées. L'API d'exportation permet à l'**Hôte ECI** de demander au **Client ECI** effectuant le

décodage d'établir une **Connexion d'exportation** avec le **Micro serveur** souhaité en vue du rechiffrement. L'API de sécurité évoluée permet au **Client ECI** exportateur d'authentifier le **Micro serveur** importateur. L'**Hôte ECI** utilise l'API d'importation pour établir la **Connexion d'exportation** autorisée avec le **Micro serveur**. L'API de rechiffrement permet à l'**Hôte ECI** d'orienter le **Micro serveur** vers un mode de fonctionnement correspondant au format de conditionnement du contenu et à l'application (streaming, visionnement différé ou stockage) et de chiffrer le contenu à l'intention du **Micro client** cible souhaité (authentifié).

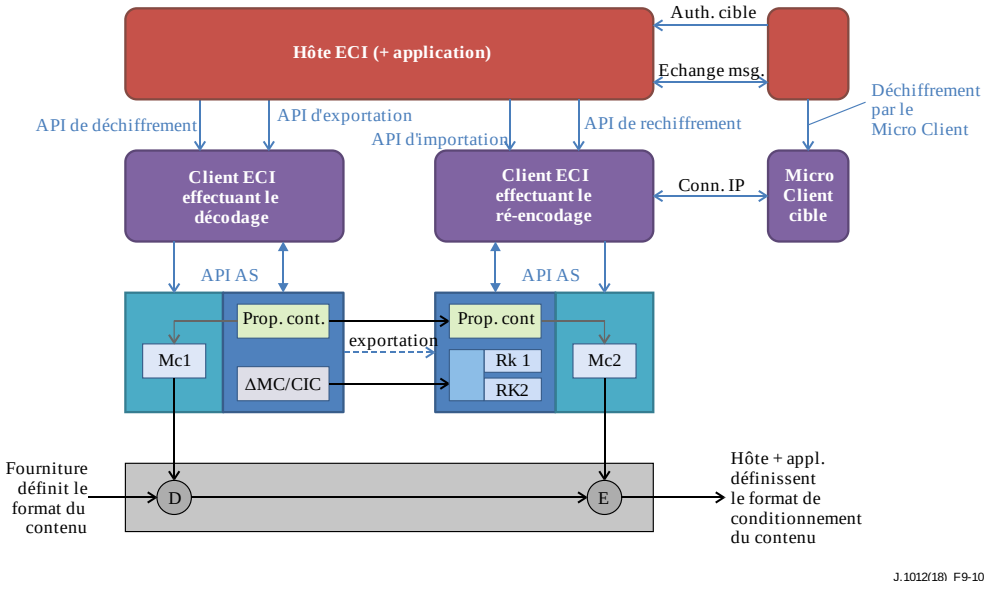


Figure 9.7.1.3-1 – Architecture des fonctionnalités de déchiffrement et de rechiffrement

Le mécanisme décrit à la Figure 9.7.1.3-1 et à la Figure 9.7.1.3-2 offre un aperçu des principaux messages des API de déchiffrement, de contrôle de l'exportation, de contrôle de l'importation, de rechiffrement, et de déchiffrement du Micro client. Il présente le flux suivi par le contenu de la gauche vers la droite, en commençant par un premier **Client ECI** CA/DRM de fourniture de contenu, suivi d'une **Connexion d'exportation/importation**, d'un **Micro serveur** qui chiffre le contenu déchiffré devant être décodé par un **Micro client cible**.

Les quatre API faisant le lien entre l'hôte et le client prennent en charge les étapes suivantes:

- L'*étape de découverte* permet aux **Clients ECI** de communiquer à l'**Hôte ECI** leurs options potentielles d'interconnexion (en collaboration avec l'application). L'**Hôte ECI** peut ainsi mettre en correspondance le contenu demandé avec un **Client ECI** en particulier. Si le **Client ECI** choisi ne dispose pas des droits requis pour traiter le contenu, l'**Hôte ECI** doit chercher d'autres **Clients ECI**. Dans les réseaux domestiques et les applications d'enregistreur vidéo personnel distribuées, cela peut impliquer des protocoles d'application tels que DLNA (voir [b-DLNA]). L'étape d'*authentification* permet à l'**Hôte ECI** d'établir une connexion authentifiée entre le **Client ECI** souhaité et le **Micro serveur**, ou entre le **Micro serveur** et le **Micro client**. L'authentification peut être implicite: la preuve du chiffrement nécessaire à l'authentification peut se traduire par la capacité du **Client ECI** à déchiffrer le contenu. L'authentification suit toujours le flux du contenu. Dans certains cas, un accord inverse est nécessaire. Pour des raisons commerciales, il pourra arriver qu'une **Connexion d'importation** requière l'approbation du **Micro serveur**.
- L'*étape d'instanciation de session* permet à l'**Hôte ECI** de réserver toutes les ressources nécessaires au déchiffrement ou au chiffrement du contenu dans un certain mode de fonctionnement associé au **Pointeur de média**. Les connexions d'importation et avec la **Cible** sont définies pour le message reqEncrMhOpen sur un **Micro serveur**, ou sont sous-entendues sur un **Client ECI** CA/DRM classique. Il convient de noter que l'**Hôte ECI** est chargé de

l'attribution de toute ressource complémentaire, telle que les ressources nécessaires au traitement de l'embrouillage, du désembrouillage, du démultiplexage et du décodage pour que l'ensemble d'un scénario d'application de média puisse se poursuivre. Enfin, le **Client ECI** demande l'affectation de ressources de sécurité évoluée et de déchiffrement ou de chiffrement au moyen de l'API de **Sécurité évoluée**.

- L'*étape de commande de session* permet à l'**Hôte ECI** de démarrer et d'arrêter le traitement du contenu au niveau des **Pointeurs de média**. Pour garantir un traitement ininterrompu du contenu sur un chemin, les **Clients ECI** doivent démarrer à partir de la destination et remonter jusqu'à la source: c'est-à-dire qu'un **Client ECI** doit être prêt à traiter le contenu lorsqu'il se présente.

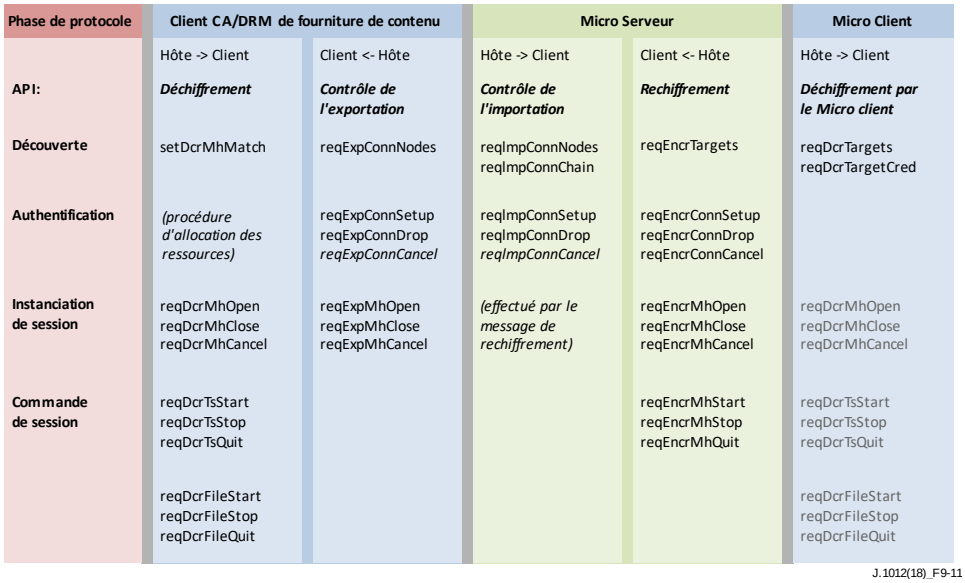


Figure 9.7.1.3-2 – Vue d'ensemble des API de chiffrement, de déchiffrement, d'importation et d'exportation

Le nom et la sémantique des messages suivent une certaine logique systématique:

- L'*étape de découverte* permet au **Client ECI** de communiquer sa capacité à se connecter à un autre **Client ECI** ou à du contenu. Les messages setDcrMhMatch, reqExpConnNodes, reqImpConnNodes, reqEncrTargets, reqDcrTargets demandent au **Client ECI** de communiquer ces capacités (sous forme d'identités).
- L'*étape d'authentification* utilise des messages se terminant par *setup*, *drop* et *cancel* pour la création d'une connexion (authentifiée), la désattribution d'une connexion précédente ou son annulation par le **Client ECI**. La connexion de référence est une **Connexion d'exportation** (**Client ECI** exportant du contenu), une **Connexion d'importation** (**Client ECI** important du contenu) ou une connexion avec une **Cible** (**Micro serveur** chiffrant du contenu pour déchiffrement ultérieur par une **Cible** et inversement, par exemple un **Micro client** déchiffrant du contenu issu d'un **Micro serveur**).
- L'*étape d'instanciation de session* utilise les termes *open*, *close* et *cancel* pour la création et l'arrêt des sessions, lesquels se réfèrent tous à un **Pointeur de média** commun. Les sessions d'interface homme-machine et la gestion des ressources liées aux **Cartes à puce**, requises par le **Client ECI**, peuvent également se référer au **Pointeur de média** pour permettre à l'**Hôte ECI** d'associer une requête de dialogue avec l'**Utilisateur** dans le contexte de l'application.
- L'*étape de commande de session* définit différents messages pour le déchiffrement de deux formats de contenu spécifiques: les flux de transport et les fichiers. Le traitement peut être démarré (*start*) et arrêté (*stop*) par l'**Hôte ECI**, et le **Client ECI** peut quitter (*quit*) le traitement en cas de ressources insuffisantes ou de problème lié aux droits.

NOTE 1 – Il est possible que dans certains systèmes de protection, il ne soit pas nécessaire de réaliser un traitement d'envergure pour toutes les phases. Les **Clients ECI** peuvent n'effectuer qu'un traitement mineur d'ordre administratif pour certains des messages.

NOTE 2 – La nature des **Clients ECI** dans une **Connexion d'importation/exportation** est différente de la relation entre un **Micro serveur** et un **Micro client**. Dans les **Connexions d'importation/exportation**, les **Clients ECI** partagent l'**Hôte ECI** et peuvent échanger du contenu par l'intermédiaire du mécanisme d'exportation AS au moyen de **Chaînes de Certificats** d'importation/exportation définies par l'interface **ECI**. Le **Micro serveur** et le **Micro client** peuvent utiliser un protocole au choix (caractéristique du **Système Micro DRM**) pour l'établissement des connexions, à condition qu'il soit adapté au cadre des API et puisse utiliser le **Système de sécurité évoluée** pour réaliser l'authentification et calculer les clés communes. Lors d'une **Connexion d'exportation/importation**, l'échange de contenu est implicite (défini par l'**Hôte ECI**); l'authenticité (à des fins d'exportation) du **Micro serveur** sera validée par le **Système de sécurité évoluée**. L'échange de contenu entre un **Micro serveur** et un **Micro client** nécessite une session de **Pointeur de média** et une commande de session aussi bien au niveau du **Micro serveur** que du **Micro client**.

9.7.2 API de contrôle de l'exportation ECI

9.7.2.1 Introduction

L'interface **ECI** permet aux **Clients ECI** d'exporter du contenu décodé vers le **Micro serveur** qui assurera le rechiffrement en vue de la redistribution (autorisée) du contenu vers d'autres dispositifs ou de son stockage (autorisé) pour lecture ultérieure. A cet effet, l'interface **ECI** définit une structure de **Certificat** établissant des groupes de **Systèmes Micro DRM** d'exportation autorisés. Chaque élément de contenu décodé est accompagné de l'identification du **Groupe d'exportation** approprié. A partir du **Groupe d'exportation**, une chaîne de **Certificats** autorisant l'exportation vers le **Micro serveur** sélectionné doit être en place. La chaîne est traitée par le **Système de sécurité évoluée** afin de fournir un mécanisme d'autorisation de l'exportation extrêmement solide.

Le **Client ECI** exportateur est chargé de fournir les **Certificats** de **Groupe d'exportation** et tous les descendants directs. Le **Micro client** effectuant l'importation est chargé de fournir les justificatifs d'identité complémentaires afin de permettre de compléter la chaîne du **Client ECI** effectuant l'exportation au **Client ECI** effectuant l'importation.

L'**Hôte ECI** peut configurer une connexion de rechiffrement entre un **Client ECI** effectuant le déchiffrement et un **Micro serveur** effectuant le chiffrement. Une fois la connexion établie, l'**Hôte ECI** peut procéder au déchiffrement et au rechiffrement du contenu au moyen de sessions de **Pointeur de média**. Le **Système de sécurité évoluée** assure la transmission sécurisée du contenu et des informations de protection associées du **Client ECI** effectuant le décodage au **Micro client** à partir des justificatifs d'identité fournis.

L'**Hôte ECI** aide les **Clients ECI** à accéder aux services de réseau afin de recevoir des justificatifs d'identité à jour pour l'exportation et l'importation, par exemple par le biais de l'API d'acquisition de carrousel de données (§ 9.5.4) et de l'API relative au protocole HTTP (§ 9.4.4.6).

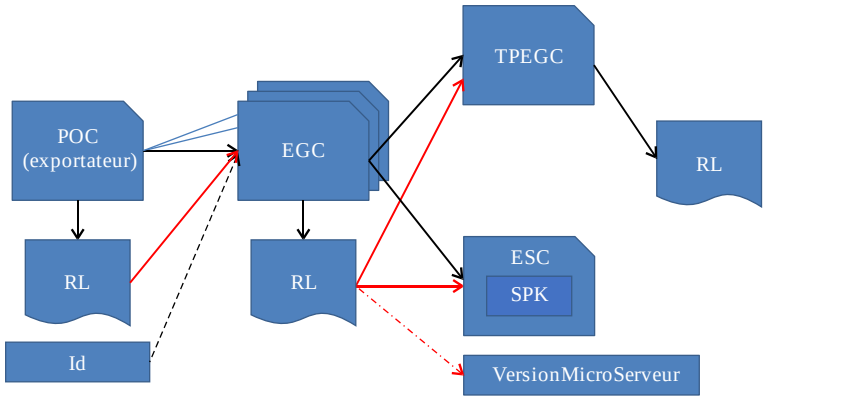
Pour cibler le rechiffrement, l'**Hôte ECI** et l'application doivent déterminer les **Micro clients** autorisés capables de décoder le contenu. Il peut s'agir aussi bien d'un **Équipement CPE** individuel (pourvu d'un Client approprié) que d'un groupe (sur la base d'une clé partagée). L'**Hôte ECI** établit ensuite une connexion autorisée entre le **Micro serveur** et le **Micro client** correspondant (une pour chaque **Micro client**). Concernant les applications de visionnement différé et d'enregistrement, les informations requises par le **Client ECI** pour décoder le contenu ultérieurement peuvent être stockées (par exemple avec le contenu rechiffré). Pour les connexions de streaming en temps réel, les messages de commande de session requis par le **Micro serveur** et le **Micro client** peuvent être transmis par l'intermédiaire de l'**Hôte ECI** dans le cas où les **Micro clients** et le **Micro Serveur** résident dans le même dispositif ou peuvent être communiqués directement entre les **Micro clients** par l'intermédiaire d'une connexion IP.

NOTE – Les protocoles de communication de **Client ECI** à **Client ECI** et les aspects connexes liés à la sécurité ne relèvent pas de l'interface **ECI**.

9.7.2.2 Structures des Certificats d'exportation

9.7.2.2.1 Structure générale

Le mécanisme d'exportation **ECI** repose sur des **Certificats**. La plupart des **Certificats** sont associés à une **Liste de révocation** afin de permettre la mise à jour des autorisations d'exportation. La Figure 9.7.2.2.1–1 présente la structure des **Certificats** visant le contrôle immédiat des exportations d'un **Client ECI** décodant du contenu.



POC: Certificat d'opération de plate-forme
RL: Liste de révocation
Id: Identificateur de Groupe d'exportation
SPK: Clé publique de l'émetteur
EGC: Certificat de Groupe d'exportation
TPEGC: Certificat de Groupe d'exportation tiers
ESC: Certificat de système d'exportation

Figure 9.7.2.2.1-1 – Structure de distribution des Certificats ECI

Le **Certificat d'opération de plate-forme** (POC) du **Client ECI** est le **Père** des **Certificats** de **Groupe d'exportation**. Le **Certificat d'opération de plate-forme ECI** dispose d'une **Liste de révocation** spéciale pour permettre au **Client ECI** de contrôler les **Certificats** de **Groupe d'exportation** et les versions des **Listes de révocation** associées. Chaque **Certificat** de **Groupe d'exportation** est le **Père** des **Certificats** d'exportation effectifs ou d'un autre **Groupe d'exportation** (descendant). Il existe deux types de **Certificats** d'exportation:

1) Un **Certificat** de système d'exportation (ESC) identifie le **Micro serveur** d'exportation autorisé au moyen de sa **Clé publique de l'émetteur**, permettant ainsi une authentification immédiate. En outre, le numéro de version de la **Liste de révocation** de l'ESC est utilisé pour définir un numéro de version minimal pour le **Micro serveur**.

2) Un **Certificat** de **Groupe d'exportation** tiers (TPEGC) correspond à un **Certificat** de **Groupe d'exportation** géré par une autre organisation. Il permet d'authentifier des groupes de **Systèmes Micro DRM** plus étendus et hétérogènes au moyen d'un seul **Certificat** d'exportation.

La structure d'un **Certificat** de **Groupe d'exportation** tiers est détaillée dans la Figure 9.7.2.2.1-2.

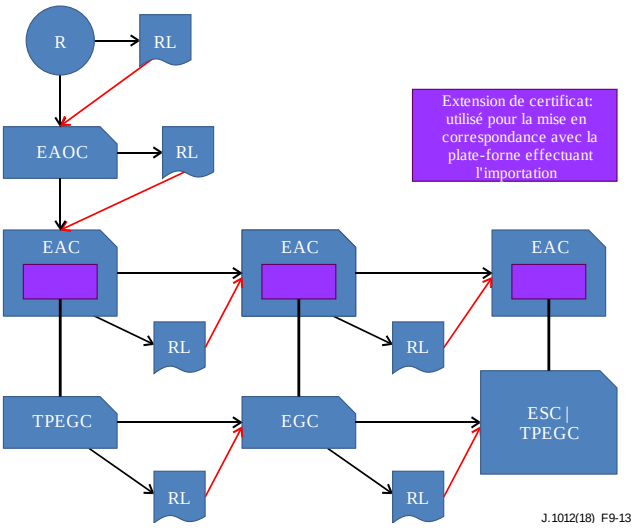


Figure 9.7.2.2.1-2 – Structure d'un Certificat de Groupe d'exportation tiers

Le **Certificat racine ECI** est le **Père** d'un **Certificat** d'opérateur d'autorisation d'exportation (EAOC), et tient à jour une **Liste de révocation** spéciale pour ces **Certificats**. Le **Certificat** d'opérateur d'autorisation d'exportation (EAOC) est le **Père** d'un **Certificat** d'autorisation d'exportation (EAC). Ce **Certificat** est mis en correspondance avec un **Certificat** de **Groupe d'exportation** tiers (TPEGC). Ce mécanisme permet ainsi la double authentification d'un groupe tiers afin de renforcer la sécurité.

Un **Certificat** de **Groupe d'exportation** tiers est le **Père**:

1) soit d'un **Certificat** de **Groupe d'exportation** (EGC), lequel peut lui-même être le **Père** d'un autre **Certificat** de **Groupe d'exportation** ou de n'importe lequel des **Certificats** répertoriés ci-dessous (chaque **Certificat** de **Groupe d'exportation** dispose d'une **Liste de révocation** associée);
2) soit d'un **Certificat** de système d'exportation (ESC);
3) soit d'un **Certificat** de **Groupe d'exportation** tiers (TPEGC) (consécutif).

Chaque **Certificat** est par ailleurs vérifié par un **Certificat** d'autorisation d'exportation (EAC) correspondant; les EAC forment une arborescence parallèlement à celle des TPEGC/EGC.

Le Tableau 9.7.2.2.1-1 fournit un aperçu des **Certificats** et de leurs **Pères**.

Tableau 9.7.2.2.1-1 – Synthèse des différents Certificats d'exportation

Nom du Certificat	Abr.	Description	Père
Groupe d'exportation	EGC	Ce **Certificat** permet aux **Clients ECI** exportateurs d'authentifier un ensemble (groupe) de **Micro clients** et/ou des groupes tiers authentifiés vers lesquels ils autorisent l'exportation. Le **Groupe d'exportation** applicable est défini comme faisant partie d'un attribut "droits authentifiés" du contenu.	POC, TPEGC, EGC
Groupe d'exportation tiers	TPEGC	**Certificat** destiné à l'authentification d'un groupe de **Systèmes Micro DRM** et géré par un tiers.	EGC, TPEGC
Opérateur d'autorisation d'exportation	EAOC	**Certificat** sur lequel se base un **Opérateur** fournissant un service d'autorisation pour des **Groupes d'exportation** tiers. Ce **Certificat** est le **Père** des arborescences de **Certificat** d'autorisation d'exportation pour les **Groupes d'exportation** tiers qu'il coauthentifie.	Racine **ECI**
Autorisation d'exportation	EAC	Ce **Certificat** assure la coauthentification d'un **Certificat** de **Groupe d'exportation** tiers ou d'un **Certificat** de **Groupe d'exportation** géré par un tiers.	EAC, EAOC
Système d'exportation	ESC	Ce **Certificat** authentifie le **Certificat d'opération de plate-forme** d'un **Micro client**.	EGC, TPEGC

9.7.2.2.2 Définitions des Certificats d'exportation

9.7.2.2.2.1 Certificat de Groupe d'exportation et Liste de révocation

La définition du **Certificat** de **Groupe d'exportation ECI** (EGC) sera conforme à la définition générale des **Certificats ECI** (ECI_certificate) figurant au § 5.2. Le **Certificat** de **Groupe d'exportation** utilise le champ d'identificateur des **Certificats ECI** avec la définition figurant dans le Tableau 9.7.2.2.2.1-1.

Tableau 9.7.2.2.2.1-1 – Définition de l'identificateur de Groupe d'exportation ECI

Syntaxe	Nbre de bits	Mnémonique
`ECI_EGC_Id {`		
`type /* voir le` Tableau 5.3-1**`*/`**	4	uimsbf
`export_group_id /* voir le` Tableau 5.3-1 **`*/`**	20	uimsbf
`export_group_version`	8	uimsbf
`}`		

Sémantique:

Type	Valeur conforme au Tableau 5.2-1.
export_group_id: nombre entier	Identificateur attribué au **Groupe d'exportation** par l'entité gérant ce dernier. Les valeurs 0x00000 et 0xFFFFF0-0xFFFFF sont réservées.
export_group_version: nombre entier	Version du **Certificat** de **Groupe d'exportation** portant l'identificateur **export_group_id**.

Afin d'assurer l'authentification des **Certificats d'Enfants**, le **Certificat** de **Groupe d'exportation** sera accompagné d'une **Liste de révocation** conformément au § 5.3, en particulier le Tableau 5.3-1.

9.7.2.2.2.2 Certificat de Groupe d'exportation tiers et Liste de révocation

La définition du **Certificat** de **Groupe d'exportation** tiers **ECI** (TPEGC) sera conforme à la définition générale des **Certificats ECI** (ECI_certificate) figurant au § 5.2. Le **Certificat** de **Groupe d'exportation** tiers utilise le champ d'identificateur des **Certificats ECI** avec la définition figurant dans le Tableau 9.7.2.2.2.2-1.

Tableau 9.7.2.2.2.2-1 – Définition du champ d'identificateur du Certificat de Groupe d'exportation tiers

Syntaxe	Nbre de bits	Mnémonique
ECI_TPEGC_Id {		
type /* voir le Tableau 5.2-1***/**	4	uimsbf
tp_export_group_id /* voir le Tableau 5.3-1 ***/**	20	uimsbf
tp_export_group_version	8	uimsbf
}		

Sémantique:

Type	Valeur conforme au Tableau 5.3-1.
tp_export_group_id: nombre entier	Identificateur attribué au **Groupe d'exportation** tiers par l'entité gérant ce dernier. Les valeurs 0x00000 et 0xFFFFF0-0xFFFFF sont réservées.
tp_export_group_version: nombre entier	Version du **Certificat** de **Groupe d'exportation** tiers portant l'identificateur **tp_export_group_id**.

Le champ d'extension du **Certificat** de **Groupe d'exportation** tiers, tel que défini dans le Tableau 9.7.2.2.2.2-2, contiendra la structure suivante reposant sur les définitions de **export_authorization_operator_id** dans le Tableau 9.7.2.2.2.4-1 et de **export_authorization_id** dans le Tableau 9.7.2.2.2.5-1.

Tableau 9.7.2.2.2.2-2 – Définition du champ d'extension du Certificat de Groupe d'exportation tiers

Syntaxe	Nbre de bits	Mnémonique
ECI_ TPEGC _Extension {		
export_authorization_operator_id	20	uimsbf
export_authorization_id	20	uimsbf
padding(4)		
Extension_Field **extension**		
}		

Sémantique:

export_authorization_operator_id: nombre entier	Identificateur **ECI** du **Certificat** d'opérateur d'autorisation d'exportation qui coauthentifie ce **Certificat**.
export_authorization_id: nombre entier	Identificateur **ECI** du **Certificat** d'autorisation d'exportation qui coauthentifie ce **Certificat** (voir le § 9.7.1.2.2.5).
extension: Extension_field	Extension de cette structure.

Afin d'assurer l'authentification des **Certificats d'Enfants**, le **Certificat** de **Groupe d'exportation** tiers sera accompagné d'une **Liste de révocation** conformément au § 5.3 et au Tableau 5.3-1.

9.7.2.2.2.3 Liste de révocation racine pour les Certificats d'Opérateur d'autorisation d'exportation

Aux fins de l'authentification, une chaîne d'authentification d'exportation doit commencer par une liste de révocation racine, conformément au § 5.3 et au Tableau 5.3-1.

9.7.2.2.2.4 Certificat d'Opérateur d'autorisation d'exportation

La définition du **Certificat** d'opérateur d'autorisation d'exportation **ECI** (EAOC) sera conforme à la définition générale des **Certificats ECI** (ECI_certificate) figurant au § 5.2. Le **Certificat** d'opérateur d'autorisation d'exportation utilise le champ d'identificateur des **Certificats ECI** avec la définition figurant dans le Tableau 9.7.2.2.2.4-1.

Tableau 9.7.2.2.2.4-1 – Définition du champ d'identificateur du Certificat d'Opérateur d'autorisation d'exportation

Syntaxe	Nbre de bits	Mnémonique
ECI_EAOC_Id {		
type /* voir le Tableau 5.3-1***/**	4	uimsbf
export_authorization_operator_id **/* voir le** Tableau 5.3-1 ***/**	20	uimsbf
export_authorization_operator_version	8	uimsbf
}		

Sémantique:

type	Valeur conforme au Tableau 5.3-1.
export_authorization_operator_id: nombre entier	Identificateur attribué à l'opérateur d'autorisation d'exportation. Les valeurs 0x00000 et 0xFFFFF0-0xFFFFF sont réservées.
export_authorization_operator_version: nombre entier	Version du **Certificat** d'opérateur d'autorisation d'exportation portant l'identificateur **export_authorization_operator_id**.

Afin d'assurer l'authentification des **Certificats d'Enfants**, le **Certificat** d'opérateur d'autorisation d'exportation sera accompagné d'une **Liste de révocation** conformément au § 5.3 et au Tableau 5.3-1.

9.7.2.2.2.5 Certificat d'autorisation d'exportation et Liste de révocation

La définition du **Certificat** d'autorisation d'exportation **ECI** (EAC) sera conforme à la définition générale des **Certificats ECI** (ECI_certificate) figurant au § 5.2 et utilisera un champ d'extension non vide spécifique. Le **Certificat** d'autorisation d'exportation utilise le champ d'identificateur des **Certificats ECI** avec la définition figurant dans le Tableau 9.7.2.2.2.5-1.

Tableau 9.7.2.2.2.5-1 – Définition du champ d'extension du Certificat d'autorisation d'exportation

Syntaxe	Nbre de bits	Mnémonique
ECI_EAC_Id {		
type /* voir le Tableau 5.3-1***/**	4	uimsbf
export_authorization_id **/* voir le** Tableau 5.3-1 ***/**	20	uimsbf
export_authorization_version	8	uimsbf
}		

Sémantique:

Type	Valeur conforme au Tableau 5.3-1.
export_authorization_id: nombre entier	Identificateur attribué au **Certificat** d'autorisation d'exportation (dans le contexte de son **Père**). Les valeurs 0x00000 et 0xFFFFF0-0xFFFFF sont réservées.
export_authorization_version: nombre entier	Version du **Certificat** d'autorisation d'exportation portant l'identificateur **export_authorization_id**.

Le champ d'extension du **Certificat** d'autorisation d'exportation contiendra la structure du **Certificat** devant être autorisée pour l'exportation (voir § 5.1.3) – à l'exception du champ **signature** –, suivie d'un champ d'extension.

Afin d'assurer l'authentification des **Certificats d'Enfants**, le **Certificat** d'autorisation d'exportation sera accompagné d'une **Liste de révocation** conformément au § 5.3 et au Tableau 5.3-1, si elle est nécessaire à l'authentification des **Certificats d'Enfants**.

9.7.2.2.2.6 Certificat de système d'exportation

La définition du **Certificat** de système d'exportation **ECI** (ESC) sera conforme à la définition générale des **Certificats ECI** (ECI_certificate) figurant au § 5.2. Le champ public_key du **Certificat** contiendra la valeur de la clé publique de l'émetteur utilisée par le **Micro serveur**. Le **Certificat** de système d'exportation utilise le champ d'identificateur des **Certificats ECI** avec la définition figurant dans le Tableau 9.7.2.2.2.6-1.

Tableau 9.7.2.2.2.6-1 – Définition du champ d'extension du Certificat de système d'exportation

Syntaxe	Nbre de bits	Mnémonique
ECI_ESC_Id {		
type /* voir le Tableau 5.3-1***/**	4	uimsbf
export_system_id **/* voir le** Tableau 5.3-1 ***/**	20	uimsbf
export_system_version	8	uimsbf
}		

Sémantique:

Type	Valeur conforme au Tableau 5.3-1.
export_system_id: nombre entier	Identificateur attribué au **Certificat** de système d'exportation (dans le contexte de son **Père**). Les valeurs 0x00000 et 0xFFFFF0-0xFFFFF sont réservées.
export_system_version: nombre entier	Version du **Certificat** de système d'exportation portant l'identificateur **export_system_id**.

9.7.2.2.3 Validation des chaînes de Certificats d'exportation

Le **Client ECI** exportateur avec une chaîne pré-validée et avec des chaînes d'autorisation d'exportation complémentaires établira la **Connexion d'importation/exportation** demandée. Le **Client ECI** exportateur et le **Micro serveur** effectuant l'importation, chacun responsable de leur partie des chaînes, fourniront à l'**Utilisateur** des informations en cas de problème et/ou de tentative d'acquisition de chaînes renouvelées. Le **Client ECI** fournira ces chaînes au **Système de sécurité évoluée** qui les traitera afin d'établir la **Connexion d'exportation/importation** souhaitée. Si le **Système de sécurité évoluée** trouve des erreurs de validation dans une chaîne ou dans l'autorisation d'exportation complémentaire, le **Client ECI** ne pourra pas établir la connexion requise.

Les **Certificats** d'autorisation d'exportation sont utilisés pour coauthentifier un **Certificat** d'exportation. Les règles de traitement en vue de la coauthentification sont les suivantes:

1) Le **Certificat** d'autorisation d'exportation et le **Certificat** à coauthentifier possèdent des signatures valides (telles que définies par leur **Père** respectif) et n'ont pas été révoqués.

2) Toutes les données du **Certificat** devant être coauthentifié, à l'exception de sa signature, sont comparées aux données du champ d'extension correspondant du **Certificat** d'autorisation d'exportation. Si elles ne correspondent pas, la coauthentification échoue.

Pour configurer la **Connexion d'exportation**, le Système de traitement des certificats suivra les règles de traitement suivantes:

1) Toutes les règles de traitement des **Chaînes de Certificats** par le Système de traitement des certificats énumérées au § 5.4.2 s'appliquent.

2) Le Système de traitement des certificats vérifiera que le type de l'**Enfant** d'un **Certificat** de **Père** est correct conformément au Tableau 5.2-2.

3) Le Père de la Chaîne d'exportation du Client ECI exportateur sera le Certificat d'opération de plate-forme ECI du Client. La Liste de révocation accompagnant les Groupes d'exportation sera appliquée pour valider les Certificats de Groupe d'exportation d'Enfants. Le numéro de

version de la Liste de révocation du Certificat d'opération de plate-forme accompagnant les Groupes d'exportation doit être supérieur à la version minClientVersion du Client (voir la Recommandation [UIT-T J.1014]).

4) Le Système de traitement des certificats acceptera au maximum deux niveaux de **Certificat** de **Groupe d'exportation** pour le **Client ECI** effectuant l'exportation, c'est-à-dire qu'un **Enfant** d'un **Certificat** de **Groupe d'exportation** de deuxième niveau sera un **Certificat** de **Groupe d'exportation** tiers ou un **Certificat** de système d'exportation.

5) Le Système de traitement des certificats s'assurera que tout **Certificat** de **Groupe d'exportation** tiers est accompagné d'un **Certificat** d'autorisation d'exportation coauthentifié par l'intermédiaire d'une chaîne (accompagnée des **Listes de révocation** pertinentes) allant de la Racine au **Certificat** d'autorisation d'exportation en passant par le **Certificat** d'opérateur d'autorisation d'exportation. La version de la liste de révocation racine pour le certificat d'opérateur d'autorisation d'exportation sera utilisée pour déterminer le numéro de version de liste de révocation maximal pour "validation de l'intégrité du système".

6) Le Système de traitement des certificats s'assurera que tout **Certificat** de **Groupe d'exportation**, de système d'exportation et de **Groupe d'exportation** tiers descendant d'un **Certificat** de **Groupe d'exportation** tiers est coauthentifié par un **Certificat** d'autorisation d'exportation qui est l'**Enfant** du **Certificat** d'autorisation d'exportation ayant validé le **Père** de ce **Certificat**.

Les **Clients ECI** effectuant l'exportation et les Serveurs **Micro DRM** doivent garantir un pré-traitement adéquat sur leurs chaînes et fournir les dernières versions disponibles afin d'éviter une révocation dans le Système de traitement des certificats.

9.7.2.2.4 Protocoles de transport pour les justificatifs d'identité d'exportation

9.7.2.2.4.1 Généralités

Les **Clients ECI** effectuant l'exportation et les **Micro serveurs** peuvent définir leur propre format pour le transport des données de justificatif d'identité. L'interface **ECI** définit un format de fichier normalisé pour le transport de ce type de données. Ces fichiers normalisés sont accessibles pour les **Clients ECI** par l'intermédiaire de l'API d'acquisition de carrousel **ECI** pour les supports de radiodiffusion. En ce qui concerne l'allocation en ligne des ressources pour les Clients, l'interface **ECI** définit à cet effet des appels d'API web normalisés.

9.7.2.2.4.2 Format du fichier d'arborescence d'exportation

Le format de fichier pour l'arborescence des **Groupes d'exportation** est défini dans le Tableau 9.7.2.2.4.2-1.

Tableau 9.7.2.2.4.2-1 – Définition du fichier d'arborescence d'exportation ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Export_Tree_File {	24	
magic = 'EET'		
image_header_version	8	uimsbf
if (image_header_version == 0x01) {		
ECI_Operator_Id **operator_id**	32	uimsbf
ECI_Platform_Operation_Id **platform_operation_id**	32	uimsbf
ECI_RL_Tree **export_group_tree**		
Extension_Field **extensions**		
}		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères 'EET'. Les **Clients ECI** contrôleront la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu et renforcer l'intégrité des données.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. Les **Clients ECI** ignoreront les images dont ils ne reconnaissent pas le numéro de version.
operator_id: ECI_Operator_Id	Identificateur de l'**Opérateur** du **Client ECI** pour l'arborescence d'exportation contenue dans le fichier. Le champ operator_version correspond à la racine du champ export_group_tree.
Platform-operation_id: ECI_Platform_Operation_Id	Identificateur de l'**Opération de plate-forme** du **Client ECI** pour l'arborescence d'exportation contenue dans le fichier.
export_group_tree: ECI_RL_Tree	La structure ECI_RL_Tree commence par la **Liste de révocation** des **Groupes d'exportation**. Pour les **Certificats** ne nécessitant pas de **Liste de révocation** complémentaire, cette structure contiendra une **Liste de révocation** vide dont la signature ne devra pas correspondre impérativement au **Certificat**.
extensions: Extension_field	Données supplémentaires telles que définies par l'opérateur.

9.7.2.2.4.3 Format du fichier de chaînes d'importation

Le format de fichier pour les **Chaînes d'importation** d'un **Micro serveur** est défini dans le Tableau 9.7.2.2.4.3-1.

Tableau 9.7.2.2.4.3-1 – Définition du format du fichier de chaînes d'importation

Syntaxe	Nbre de bits	Mnémonique
ECI_Import_Chain_File {	24	
magic = 'EIC'		
image_header_version	8	uimsbf
if (image_header_version == 0x01) {		
ECI_Operator_Id **operator_id**	32	uimsbf
ECI_Platform_Operation_Id **platform_operation_id**	32	uimsbf
nr_chains	16	uimsbf
padding(4)		
for (i=0; i<nr_chains; i++){		
ECI_Operator_Id **eaoc_id**	32	uimsbf
ECI_Platform_Operation_Id **eac_id**	32	uimsbf
ECI_certificate_Chain **import_chain**		
}		
Extension_Field **extensions**		
}		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères 'EIC'. Les **Clients ECI** contrôleront la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu et renforcer l'intégrité des données.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. Les **Clients ECI** ignoreront les images dont ils ne reconnaissent pas le numéro de version.
operator_id: ECI_Operator_Id	Identificateur de l'**Opérateur** du **Micro serveur** pour lequel cette **Chaîne d'importation** est prévue.
platform_operation_id: ECI_Platform_Operation_Id	Identificateur de l'**Opération de plate-forme** du **Micro serveur** pour lequel cette **Chaîne d'importation** est prévue.
nr_chains: nombre entier	Nombre de **Chaînes d'importation** contenues dans le fichier.
eaoc_id: ECI_Operator_Id	Identificateur de l'**Opérateur** d'autorisation de la **Chaîne d'importation**.
eac_id: ECI_Platform_Id	Identificateur du **Certificat** d'autorisation d'exportation autorisant l'**Opération de plate-forme** de la **Chaîne d'importation**.
import_chain: ECI_certificate_Chain	**Chaîne de certificats ECI** depuis le **Certificat d'opération de plate-forme** d'importation jusqu'au **Certificat** de système d'exportation identifiant le **Micro client**. La chaîne pourra contenir plusieurs **Certificats** de **Groupe d'exportation** tiers. Chaque **Chaîne d'importation** valide sera représentée séparément, c'est-à-dire que si la chaîne1 est composée de deux sous-chaînes tierces et que la deuxième sous-chaîne peut également être utilisée séparément comme **Chaîne d'importation**, elle sera représentée séparément. Pour les **Certificats** ne nécessitant pas de **Liste de révocation** complémentaire, cette structure contiendra une **Liste de révocation** vide dont la signature ne devra pas correspondre impérativement au **Certificat**.
extensions: Extension_field	Données supplémentaires telles que définies par l'opérateur.

9.7.2.2.4.4 Format du fichier d'autorisation d'exportation

Le format de fichier pour l'autorisation des **Chaînes d'exportation** d'un **Micro serveur** est défini dans le Tableau 9.7.2.2.4.4-1.

Tableau 9.7.2.2.4.4-1 – Définition du fichier d'autorisation d'exportation ECI

Syntaxe	Nbre de bits	Mnémonique
ECI_Export_Authorization_File {	24	
magic = 'EEA'		
image_header_version	8	uimsbf
if (image_header_version == 0x01) {		
ECI_Operator_Id **operator_id**	32	uimsbf
ECI_Platform_Operation_Id **platform_operation_id**	32	uimsbf
nr_chains	16	uimsbf
padding(4)		
for (i=0; i<nr_chains; i++){		
direct_flag	1	uimsbf
padding(4)		
ECI_Operator_Id **o_id**	32	uimsbf
ECI_Platform_Operation_Id **po_id**	32	uimsbf
ECI_certificate_Chain **chain**		
}		
Extension_Field **extensions**		
}		
}		

Sémantique:

magic: octet[3]	Nombre magique servant à vérifier le format des données suivantes. Il a la valeur de trois représentations 8 bits ASCII des caractères 'EEA'. Les **Clients ECI** contrôleront la valeur de ce champ pour vérifier si un fichier **ECI** possède le format attendu et renforcer l'intégrité des données.
image_header_version: octet	Version du format de l'en-tête d'image. La valeur 0x01 est celle actuellement définie pour cette version. Toutes les autres valeurs sont réservées. Les **Clients ECI** ignoreront les images dont ils ne reconnaissent pas le numéro de version.
operator_id: ECI_Operator_Id	Identificateur de l'**Opérateur** du **Micro serveur** pour lequel cette **Chaîne d'importation** est prévue.
Platform_operation_id: ECI_Platform_Operation_Id	Identificateur de l'**Opération de plate-forme** du **Micro serveur** pour lequel cette **Chaîne d'importation** est prévue.
nr_chains: nombre entier	Nombre de chaînes d'autorisation d'exportation dans ce fichier.
direct_flag: bit	Si la valeur est 0b1, la chaîne suivante autorise directement une sous-chaîne de **Certificat** de système d'exportation, et **o_id** et **po_id** ne sont pas à prendre en compte. Si la valeur est 0b0, la chaîne suivante autorise une sous-chaîne de **Certificat** de Groupe d'exportation tiers, et **o_id** et **po_id** représentent des identificateurs de **Certificat** d'autorisation.
o_id: ECI_Operator_Id	Identificateur de l'**Opérateur** tiers et de la **Chaîne d'exportation** tierce temporaire authentifiés par la chaîne d'authentification d'exportation suivante.
po_id: ECI_Platform_Operation_Id	Identificateur de l'**Opération de plate-forme** tierce et de la **Chaîne d'exportation** tierce temporaire authentifiées par la chaîne d'authentification d'exportation suivante.

chain: ECI_certificate_Chain	Chaîne de certificats ECI du **Certificat racine ECI** au **Certificat** d'autorisation d'exportation authentifiant le premier **Certificat** de **Groupe d'exportation** tiers ou **Certificat** de système d'exportation.
extensions: Extension_field	Données supplémentaires telles que définies par l'opérateur.

9.7.2.2.4.5 Carrousels de radiodiffusion transportant des justificatifs d'identité d'exportation

Les **Opérateurs** peuvent déployer des carrousels de type **ECI** tels que définis au § 7.7.2 pour transporter les justificatifs d'identité d'exportation et/ou d'importation des **Clients ECI** qu'ils choisissent de prendre en charge. Cependant, pour un **Client ECI** spécifique, l'**Hôte ECI** n'aura à surveiller que les mises à jour d'un unique message DSI d'emplacement d'un carrousel de données. Cela signifie que pour transporter les justificatifs d'identité d'exportation et d'importation via le format de carrousel standard, l'**Opérateur** utilisera le même carrousel que celui transportant l'image du Client, les justificatifs d'identité de l'**Opération de plate-forme** et les données de révocation, etc. pour ce **Client ECI**. Voir également le § 7.7.2.1.

Le format des données des modules de carrousel doit être conforme au Tableau 7.7.2.6–1. Les modules désignés par un descripteur de compatibilité compatibilityDescriptor dont le champ descriptorType est égal à 0xB0 transporteront des modules comportant une seule structure ECI_Export_Tree_File, ceux dont le champ descriptorType est égal à 0xB1 transporteront des modules comportant une seule structure ECI_Import_Chain_File et ceux dont le champ descriptorType est égal à 0xB2 transporteront des modules comportant une seule structure ECI_Export_Authentication_File.

Il est recommandé que le suivi par le **Client ECI** des mises à jour dans le carrousel coïncide avec celles devant être réalisées par l'**Hôte ECI** pour les autres données de **Client ECI** afin de garantir une gestion efficace de l'alimentation.

9.7.2.2.4.6 Fourniture en ligne des justificatifs d'identité d'exportation

La présente Recommandation réserve les structures URL d'API web suivantes afin de fournir une structure standard permettant aux **Clients ECI** d'accéder aux justificatifs d'identité d'exportation depuis un serveur en ligne de l'**Opérateur**.

Voir le § 7.7.3 pour la définition de tail_extension et les conventions de notation:

```
tail_extension* ::=
         client_export |
         client_import |
         client_exp_auth .
```

La notation tail_extension* indique que d'autres extensions peuvent être présentes dans des versions ultérieures de la présente Recommandation.

Les **Requêtes** d'API web suivantes sont définies pour l'importation/exportation.

```
client_export ::= 'client-export/' operator_id '/' platform_operation_id .
```

Cette **Requête** renvoie la dernière version du fichier d'arborescence d'exportation au format ECI_Export_Tree_File pour le **Client ECI** désigné par **operator_id**, **platform_operation_id**.

```
client_import ::= 'client-import/' operator_id '/' platform_operation_id .
```

Cette **Requête** renvoie la dernière version du fichier de **chaîne d'importation** au format ECI_Import_Chain_File pour le **Micro serveur** désigné par **operator_id**, **platform_operation_id**.

```
client_exp_auth ::= 'client-exp_auth/' operator_id '/' platform_operation_id .
```

Cette **Requête** renvoie la dernière version du fichier d'authentification d'exportation au format ECI_Export_Authentication_File pour le **Micro serveur** désigné par **operator_id**, **platform_operation_id**.

9.7.2.3 API de Connexion d'exportation

9.7.2.3.1 Généralités

Les **Clients ECI** peuvent fournir des informations d'exportation à l'**Hôte ECI**. Cela permet à l'**Hôte ECI** d'associer le système exportateur aux **Chaînes d'importation** correspondantes des **Micro serveurs**. L'**Hôte ECI** (et donc l'application) peut définir les connexions à établir effectivement à partir de toutes les options possibles. Il peut tenter de connecter le **Client ECI** exportateur et le **Client ECI** importateur correspondant en envoyant au premier une demande de connexion avec la **Chaîne d'importation** pour le **Client ECI** importateur cible. Le **Client ECI** exportateur et l'**Hôte ECI** peuvent demander à annuler la connexion, ou à la réinitialiser si les justificatifs d'identité d'importation ont été mis à jour. Les messages de **Connexion d'exportation** disponibles sont répertoriés dans le Tableau 9.7.2.3.1-1.

Tableau 9.7.2.3.1-1 – Messages de l'API de Connexion d'exportation

Message	Type	Sens	Étiquette	Description
reqExpConnNodes	A	H→C	0x0	L'**Hôte ECI** demande les nœuds des options d'exportation au **Client ECI**.
reqExpConnSetup	A	H→C	0x1	L'**Hôte ECI** demande au **Client ECI** d'initialiser une **Connexion d'exportation** vers un **Client ECI** importateur sur la base d'une **Chaîne d'importation**.
reqExpConnDrop	A	H→C	0x2	L'**Hôte ECI** annule toute connexion initialisée précédemment entre un **Client ECI** exportateur et un **Client ECI** importateur.
reqExpConnCancel	A	C→H	0x3	Le **Client ECI** met fin à une **Connexion d'exportation** initialisée avec un **Client ECI** importateur.
reqExpMhOpen	A	H→C	0x4	L'**Hôte ECI** demande au **Client ECI** de créer une session d'exportation à partir d'une **Connexion d'exportation** initialisée précédemment.
reqExpMhClose	A	H→C	0x5	L'**Hôte ECI** ferme une session d'exportation.
reqExpMhCancel	A	C→H	0x6	Le **Client ECI** annule une session d'exportation.

9.7.2.3.2 Message reqExpConnNodes

H→C reqExpConnNodes() →
C→H resExpConnNodes(ExpConnOption **conn Nodes [])**

- Le message demande au **Client ECI** de renvoyer sa liste de **Connexions d'exportation** possibles; le message de **Réponse** renvoie la liste. Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.3.2-2.

Définitions des paramètres de la réponse:

connNodes: ExpConn Option[]	La liste fournit les identités **ECI** de clients tiers ou de **Clients ECI** avec lesquels le **Client ECI** peut se connecter en vue de l'exportation. Chaque option est assortie d'une priorité: plus la priorité est élevée, plus le risque que l'exportation échoue est faible. ExpConnNode est défini dans le Tableau 9.7.2.3.2–1.

Tableau 9.7.2.3.2-1 – Définition du type ExpConnNode

```
typedef struct ExpConnNode {
   uint  targetType;
   uint  operatorId;
   uint  targetId;
   uint  targetPriority;
} ExpConnNode;
```

Définitions des champs:

targetType: uint	Type de cible: **Certificat** d'autorisation d'exportation si la valeur est égale à 1 (tiers), **Certificat** d'opération de plate-forme si la valeur est égale à 2 (exportation directe). Les autres valeurs ne sont pas définies.
operatorId: uint	Représente l'identificateur sur 20 bits du **Certificat ECI** de l'**Opérateur** de l'exportation vers la cible: export_authorization_operator_id pour la cible du **Certificat** d'autorisation d'exportation et operator_id pour la cible du **Certificat** d'opération de plate-forme.
targetId: uint	Représente l'identificateur sur 20 bits du **Certificat ECI** de l'exportation vers la cible: export_authorization_id pour la cible du **Certificat** d'autorisation d'exportation et platform_operation_id pour la cible du **Certificat** d'opération de plate-forme.
targetPriority: uint	La priorité déterminant la sélection d'une exportation particulière est la somme de deux éléments: • Valeur exprimée en multiples de 1 024 représentant une priorité (commerciale) spécifique pour l'exportation devant être connectée à un **Micro serveur** en particulier. • Valeur comprise entre 0 et 1 023 représentant une fraction moins 1 de 1 024 pour les cas d'utilisation prévus du contenu pouvant être exporté avec ce **Système Micro DRM** d'exportation. Les **Hôtes ECI** utiliseront ces informations pour sélectionner automatiquement le **Système Micro DRM** le plus approprié (à condition que le système ayant la priorité la plus élevée respecte les exigences relatives à l'application Micro DRM) et/ou pour le présenter comme préférence à l'**Utilisateur** en cas de sélection manuelle.

Tableau 9.7.2.3.2-2 – Codes d'erreur du message reqExpConnNodes

Nom	Description
ErrExpConnNwAccess	Voir le Tableau 9.7.2.3.9-1.
ErrExpConnAuthProblem	
ErrExpUninitState	

9.7.2.3.3 Message reqExpConnSetup

H→C reqExpConnSetup (CertChainSerial **Import,** CertChainSerial **Auth[],**ushort **connId)** → **C→H resExpConnSetup ()**

- Ce message demande au **Client ECI** d'initialiser (ou de réinitialiser) une **Connexion d'exportation connId** avec le **Client ECI** doté de l'identificateur **clientId** au moyen de la **Chaîne d'importation Import**, des chaînes d'authentification d'exportation **Auth** et de la **Cible** de la chaîne du **Client ECI**.

Définitions des paramètres de la requête:

Import: CertChainSerial	**Chaîne d'importation** (du **Certificat** de **Groupe d'exportation** tiers d'exportation au **Certificat** de système d'exportation).
Auth: CertChainSerial[]	Chaînes d'authentification d'exportation depuis la **Racine** jusqu'au **Certificat** d'autorisation d'exportation authentifiant le premier **Certificat** de Groupe d'exportation tiers dans une sous-chaîne de tiers unique. Les chaînes du paramètre **Auth** se présentent dans l'ordre, du **Certificat** de Groupe d'exportation tiers exportateur établissant la connexion au **Certificat** d'opération de plate-forme importateur.
connId: ushort	Identificateur de la **Connexion d'exportation** attribué par l'**Hôte ECI**.

Définition du type et du type de tableau CertChainSerial

CertChainSerial est la représentation dans l'ordre des octets du réseau (big endian) de la fonction `ECI_certificate_Chain` définie dans le Tableau 5.4.1-1, avec un remplissage par un multiple de 32 bits.

CertChainSerial[] est défini par la structure de données (de type C) suivante:

```
typedef struct CertChainSerial {
   uint  numberElements;    /* nombre d'éléments dans le tableau de
chaînes*/
   uint  elementIndex[];    /* index du début de chaque élément dans
le conteneur de données chainElements */
   uint  chainElements[];   /* conteneur de données incluant le nombre
numberElements
                             de représentations CertChainSerial des
                             chaînes successives du tableau. */
  } CertChainSerial;
```

elementIndex et chainElements seront représentés en tant que tableau de données inline dans la structure de données certChainSerialArray.

Sémantique détaillée:

- Les **Hôtes ECI** peuvent émettre une requête reqExpConnSetup pour le compte d'une connexion existante afin d'informer le **Client ECI** exportateur que le **Client ECI** importateur a (potentiellement) de nouveaux justificatifs d'identité d'importation. Sauf si la connexion en cours peut être abandonnée immédiatement, il est recommandé que les **Clients ECI** exportateurs reportent le renouvellement de la connexion avec le **Client ECI** importateur jusqu'à ce qu'il n'y ait plus de session active.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.3.3-1.

Tableau 9.7.2.3.3-1 – Codes d'erreur du message reqExpConnSetup

Nom	Description
ErrExpConnNwAccess	Voir le Tableau 9.7.2.3.9-1.
ErrExpConnAuthProblem	
ErrExpUninitState	
ErrExpInvalidChain	

9.7.2.3.4 Message reqExpConnDrop

H→C reqExpConnDrop(ushort **connId)** →
C→H resExpConnDrop()

- Ce message demande au **Client ECI** d'abandonner une **Connexion d'exportation** avec le client identifiée par **connId**.

Définitions des paramètres de la requête:

connId: ushort	Identificateur de la **Connexion d'exportation**.

Préconditions de la requête:

1) Une **Connexion d'exportation** (identifiée par **connId**) a été précédemment établie.

Postconditions de la réponse:

1) La **Connexion d'exportation** (le cas échéant) est fermée.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.3.4-1.

Tableau 9.7.2.3.4-1 – Codes d'erreur du message reqExpConnDrop

Nom	Description
ErrExpConnNone	Voir le Tableau 9.7.2.3.9-1.

9.7.2.3.5 Message reqExpConnCancel

C→H reqExpConnCancel(ushort **connId)** →
H→C resExpConnCancel()

- Ce message informe l'**Hôte ECI** que le **Client ECI** a mis fin à la **Connexion d'exportation** identifiée par **connId**.

Définitions des paramètres de la requête:

connId: ushort	Identificateur attribué à la connexion.

Préconditions de la requête:

1) Une **Connexion d'exportation** identifiée par **connId** a été précédemment établie.

9.7.2.3.6 Message reqExpMhOpen

H→C reqExpMhOpen(ushort **mhExp,** ushort **mhDcr,** ushort **connId)** →
C→H resExpMhOpen(ushort **mhExp)**

- Ce message demande au **Client ECI** de créer une session d'exportation identifiée par le **Pointeur de média mh** à partir de la **Connexion d'exportation connId**.

Définitions des paramètres de la requête:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.
mhDcr: ushort	**Pointeur de média** de la session de déchiffrement à exporter.
connId: ushort	Identificateur attribué à la **Connexion d'exportation**.

Définitions des paramètres de la réponse:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.

Préconditions de la requête:

1) Une **Connexion d'exportation connId** a été précédemment établie.
2) Une session de déchiffrement **mhDcr** a été précédemment établie.

Postconditions de la requête:

1) Une **Connexion d'exportation** est établie ou une erreur s'est produite.

Sémantique détaillée:

- Le **Client ECI** exportateur peut suspendre et reprendre une exportation sur une session existante, par exemple sur la base de l'inclusion de la connexion dans le **Groupe d'exportation**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.3.6-1.

Tableau 9.7.2.3.6-1 – Codes d'erreur du message reqExpMhOpen

Nom	Description
ErrExpConnNone	Voir le Tableau 9.7.2.3.9-1.
ErrExpDcrMhNone	

9.7.2.3.7 Message reqExpMhClose

H→C reqExpMhClose(ushort **mhExp) →**
C→H resExpMhClose(ushort **mhExp)**

- Ce message demande au **Client ECI** de fermer une session d'exportation identifiée par le **Pointeur de média mh** à partir de la **Connexion d'exportation connId**.

Définitions des paramètres de la requête:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.

Définitions des paramètres de la réponse:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.

Préconditions de la requête:

1) Une session d'exportation **mhExp** a été précédemment établie et n'est pas encore terminée.

Postconditions de la requête:

1) La session d'exportation **mhExp** est arrêtée.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.3.7-1.

Tableau 9.7.2.3.7-1 – Codes d'erreur du message reqExpMhClose

Nom	Description
ErrExpMhNone	Voir le Tableau 9.7.2.3.9-1.

9.7.2.3.8 Message reqExpMhCancel

C→H reqExpMhCancel(ushort **mhExp) →**
H→C resExpMhCancel(ushort **mhExp)**

Ce message informe l'**Hôte ECI** que le **Client ECI** a arrêté la session d'exportation **mhExp**.

Définitions des paramètres de la requête:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.

Définitions des paramètres de la réponse:

mhExp: ushort	**Pointeur de média** attribué par l'**Hôte ECI** à la **Connexion d'exportation**.

Préconditions de la requête:

1) Une session d'exportation **mhExp** a été précédemment établie.
2) Le **Client ECI** a mis fin à la session.

9.7.2.3.9 Codes d'erreur de l'API de Connexion d'exportation

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont définies dans le Tableau 9.7.2.3.9-1.

Tableau 9.7.2.3.9-1 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrExpConnNwAccess	-256	L'accès au réseau fournissant les informations demandées est impossible ou est plus lent que prévu et n'a pu aboutir.
ErrErrConnAuthProblem	-257	Des incohérences internes ont été détectées dans les données fournies et ont empêché la **Requête** d'aboutir.
ErrExpConnUninitState	-258	Le **Client ECI** a d'abord besoin de ressources et/ou d'autres fonctions d'exécution pour pouvoir répondre à la **Requête**.
ErrExpConnInvalidChain	-259	Une chaîne fournie au **Client ECI** est non valide et/ou n'a pas pu être authentifiée au moyen des chaînes d'authentification.
ErrExpConnNone	-260	La connexion n'existe pas.
ErrExpMhNone	-261	La session d'exportation indiquée par le **Pointeur de média** n'est pas prise en charge par le **Client ECI**.
ErrExpDcrMhNone	-262	La session de déchiffrement indiquée par le **Pointeur de média** n'est pas prise en charge par le **Client ECI**.

9.7.2.4 API de Connexion d'importation

9.7.2.4.1 Généralités

Les **Clients ECI** peuvent fournir leurs **Chaînes d'importation** à l'**Hôte ECI**. Cela permet à l'**Hôte ECI** de connecter le **Client ECI** importateur aux options d'exportation correspondantes des **Micro serveurs**. L'**Hôte ECI** et l'application peuvent choisir les connexions à établir parmi les options disponibles. L'**Hôte ECI** peut commencer à configurer une connexion entre le **Client ECI** exportateur et le **Client ECI** importateur en demandant tout d'abord au Client importateur l'autorisation de le connecter au **Client ECI** exportateur. Le Client importateur peut refuser cette connexion par exemple pour des raisons d'ordre commercial liées à son **Opérateur**. Si une connexion est établie, le **Client ECI** importateur et l'**Hôte ECI** peuvent demander à l'annuler, ou à la réinitialiser si les justificatifs d'identité d'importation ont été mis à jour.

Les chaînes d'entrée sont identifiées par leur premier nœud, à savoir les identificateurs **ECI** du **Certificat** d'opérateur d'autorisation d'exportation et du **Certificat** d'autorisation d'exportation pour le **Certificat** de Groupe d'exportation tiers. Ce nœud est désigné dans le Tableau 9.7.2.4.1-1 par le terme *nœud d'importation*.

Tableau 9.7.2.4.1-1 – Messages de l'API de Connexion d'importation

Message	Type	Sens	Étiquette	Description
reqImpConnNodes	A	H→C	0x0	L'**Hôte ECI** demande au **Client ECI** importateur de fournir ses nœuds d'importation.
reqImpConnChain	A	H→C	0x1	L'**Hôte ECI** demande au **Client ECI** importateur de fournir la chaîne d'entrée d'un nœud d'importation spécifique.
reqImpConnChainRenew	A	C→H	0x2	Le **Client ECI** demande à l'**Hôte ECI** de réinitialiser la connexion avec une **Chaîne d'importation** mise à jour.
reqImpConnSetup	A	H→C	0x3	L'**Hôte ECI** demande au **Client ECI** importateur d'initialiser une **Connexion d'importation** avec un **Client ECI** exportateur spécifique par l'intermédiaire d'un nœud d'importation.
reqImpConnDrop	A	H→C	0x4	L'**Hôte ECI** abandonne la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.
reqImpConnCancel	A	C→H	0x5	Le **Client ECI** met fin à la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.

9.7.2.4.2 Message reqImpConnNodes

H→C reqImpConnNodes () →
C→H resImpConnNodes(ImpConnNode **nodes[])**

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** importateur de fournir ses nœuds d'importation.

Définitions des paramètres de la réponse:

nodes[]: ImpConnNode	Tableau des nœuds d'importation et du nombre d'intermédiaires tiers. La structure de ImpConnNodes est définie dans le Tableau 9.7.2.4.2–1.

Tableau 9.7.2.4.2-1 – Définition du type ImpConnNode

```
typedef str uct ImpConnNode {
   uint  targetType;
   uint  operatorId;
   uint  targetId;
   uint  intermediaries
} ImpConnNode;
```

Définitions des champs:

targetType: uint	Type de cible: **Certificat** d'autorisation d'exportation si la valeur est égale à 1 (tiers), **Certificat** d'opération de plate-forme si la valeur est égale à 2 (exportation directe). Les autres valeurs ne sont pas définies.
operatorId: uint	Représente l'identificateur sur 20 bits du **Certificat ECI** de l'**Opérateur** de l'importation vers la **Cible**: export_authorization_operator_id pour la cible du **Certificat** d'autorisation d'exportation ou operator_id pour la cible du **Certificat** d'opération de plate-forme.
targetId: uint	Représente l'identificateur sur 20 bits du **Certificat ECI** de l'importation vers la cible: export_authorization_id pour la cible du **Certificat** d'autorisation d'exportation ou platform_operation_id pour la cible du **Certificat** d'opération de plate-forme.
intermediaries: uint	Représente le nombre d'intermédiaires tiers depuis le nœud d'entrée jusqu'au **Certificat** d'opération de plate-forme du **Client ECI** importateur. L'**Hôte ECI** sélectionnera la **Chaîne d'importation** la plus courte parmi les options d'exportation ayant la même priorité targetPriority pour le **Client ECI** exportateur.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.4.2-2.

Tableau 9.7.2.4.2-2 – Codes d'erreur du message reqImpConnNodes

Nom	Description
ErrImpConnNwAccess	Voir le Tableau 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpUninitState	

9.7.2.4.3 Messages reqImpConnChain et reqImpConnChainRenew

H→C reqImpConnChain(ImpConnNode **node)** →
C→H resImpConnChain(CertChainSerial **Import,** CertChainSerial **Auth[])**

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** importateur de fournir la chaîne d'entrée d'un nœud d'importation spécifique.

C→H reqImpConnChainRenew(CertChainSerial **Import,** CertChainSerial**Auth[])** →
H→C resImpConnChainRenew()

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** de réinitialiser la connexion avec une **Chaîne d'importation** mise à jour.

Paramètre de la Requête pour reqImpConnChain:

node: ImpConnNode	Noeud d'importation pour lequel la **Chaîne d'importation** sera renvoyée à l'**Hôte ECI**.

Définitions des paramètres de la requête pour reqImpConnChainRenew et définitions des paramètres de la réponse pour reqImpConnChain:

Import: CertChainSerial	**Chaîne d'importation** (du **Certificat** de **Groupe d'exportation** tiers d'exportation au **Certificat** de système d'exportation).
Auth: CertChainSerial[]	Chaînes d'authentification d'exportation depuis la **Racine** jusqu'au **Certificat** d'autorisation d'exportation authentifiant le premier **Certificat** de Groupe d'exportation tiers dans une sous-chaîne de tiers unique. Les chaînes du paramètre **Auth** se présentent dans l'ordre, du **Certificat** de Groupe d'exportation tiers exportateur au **Certificat** d'opération de plate-forme importateur.

Préconditions de la requête reqImpConnChainRenew:

1) Une **Connexion d'importation** a été établie précédemment avec un **Client ECI** au moyen d'un élément de la chaîne fournie.

Sémantique détaillée pour reqImpConnChainRenew:

- L'**Hôte ECI** transmettra immédiatement les informations relatives à la chaîne mise à jour aux **Clients ECI** exportateurs concernés.
- Il est recommandé que les opérateurs fournissent les chaînes mises à jour bien avant que la chaîne précédente ne devienne obsolète, afin de garantir un service ininterrompu.

Les codes d'erreur associés au message reqImpConnChain sont répertoriés dans le Tableau 9.7.2.4.3-1.

Tableau 9.7.2.4.3-1 – Codes d'erreur du message reqImpConnChain

Nom	Description
ErrImpConnNwAccess	Voir le Tableau 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	

Les codes d'erreur associés au message reqImpConnChainRenew sont répertoriés dans le Tableau 9.7.2.4.3-2.

Tableau 9.7.2.4.3-2 – Codes d'erreur du message reqImpConnChainRenew

Nom	Description
ErrImpConnNoConn	Voir le Tableau 9.7.2.4.7-1.

9.7.2.4.4 Message reqImpConnSetup

H→C reqImpConnStart (ImpConnNode **node,** ushort **exportClientId,** ushort **connId)** → **C→H resImpConnStart(**)

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** importateur d'établir une **Connexion d'importation** avec un **Client ECI** exportateur spécifique par l'intermédiaire d'un nœud d'importation.

Paramètres de la Requête:

node: ImpConnNode	Noeud d'importation par l'intermédiaire duquel la connexion est établie.
exportClientId: ushort	Identification par l'**Hôte ECI** du **Client ECI** exportateur.
connId: ushort	Identificateur attribué à la **Connexion d'importation**.

Sémantique détaillée:

- Le **Client ECI** peut rejeter la **Connexion d'importation** pour des raisons d'ordre commercial liées à son **Opérateur**.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.4.4-1.

Tableau 9.7.2.4.4-1 – Codes d'erreur du message reqImpConnSetup

Nom	Description
ErrImpConnNwAccess	Voir le Tableau 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnRefuseComm	
ErrImpConnUnknError	

9.7.2.4.5 Message reqImpConnDrop

H→C reqImpConnDrop (ushort **connId)** →
**C→H resImpConnDrop(**)

- Ce message permet à l'**Hôte ECI** d'abandonner la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.

Paramètres de la Requête:

connId: ushort	Identification par l'**Hôte ECI** de la **Connexion d'importation** à abandonner.

Préconditions de la requête:

1) Une **Connexion d'importation** (identifiée par **connId**) a été précédemment initialisée.

Postconditions de la réponse:

1) La **Connexion d'exportation** (le cas échéant) est fermée.

Les codes d'erreur associés sont répertoriés dans le Tableau 9.7.2.4.5-1.

Tableau 9.7.2.4.5-1 – Codes d'erreur du message reqImpConnDrop

Nom	Description
ErrImpConnNwAccess	Voir le Tableau 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnNoConn	

9.7.2.4.6 Message reqImpConnCancel

C→H reqImpConnCancel (ushort **connId)**→
**H→C resImpConnCancel(**)

- Ce message permet au **Client ECI** de mettre fin à la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.

Paramètres de la Requête:

connId: ushort	Une **Connexion d'importation** (identifiée par **connId**) a été précédemment initialisée.

Préconditions de la requête:

1) Une **Connexion d'importation** a été précédemment établie avec le Client doté de l'identificateur de client fourni par l'**Hôte ECI exportClientId**, et est fermée.

9.7.2.4.7 Codes d'erreur de l'API de Connexion d'importation

Les valeurs des erreurs spécifiques à l'API pouvant être renvoyées par les messages de **Réponse** de cette API sont répertoriées dans le Tableau 9.7.2.4.7-1.

Tableau 9.7.2.4.7-1 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrImpConnNwAccess	-256	L'accès au réseau fournissant les informations demandées était plus lent que prévu.
ErrImpConnAuthProblem	-257	Des incohérences internes ont été détectées dans les données fournies et ont empêché la **Requête** d'aboutir.
ErrImpUninitState	-258	Le **Client ECI** a d'abord besoin de ressources et/ou d'autres fonctions d'exécution pour pouvoir répondre à la **Requête**.
ErrImpConnRefuseComm	-259	Une chaîne fournie au **Client ECI** est non valide et/ou n'a pas pu être authentifiée au moyen des chaînes d'authentification.
ErrImpConnRefuseComm	-260	Le **Client ECI** importateur refuse la connexion au **Client ECI** exportateur pour des raisons d'ordre commercial.
ErrImpConnUnknError	-261	Le **Client ECI** importateur a rencontré une erreur inconnue.
ErrExpConnNone	-262	La connexion n'existe pas.

9.7.2.5 API de rechiffrement

9.7.2.5.1 Généralités

L'API de rechiffrement permet à un **Micro serveur** de rechiffrer le contenu provenant d'une **Connexion d'importation** spécifique à un client issu d'un groupe, en vue de son décodage ultérieur par un **Micro client**. Il se peut que le décodage doive être réalisé de manière quasi instantanée (connexion de type streaming) et ne permette pas de lire à nouveau le contenu lors d'une session ultérieure; le contenu rechiffré peut également être stocké ou visionné en différé avec les informations de déchiffrement associées à destination du **Micro client** effectuant le décodage, pour être décodé à un stade ultérieur.

La phase de découverte permet à l'application de mettre en correspondance un **Micro serveur** avec une **Cible** possible (**Micro client** ou groupe de **Micro clients**) et d'échanger les informations d'authentification nécessaires du **Micro client** vers le **Micro serveur** en vue de l'authentification du **Micro client** et d'un transfert sécurisé de contenu. L'**Hôte ECI** peut choisir un mode de communication bidirectionnel (basé sur IP ou messages passant par l'intermédiaire de l'**Hôte ECI**) afin de permettre la prise en charge de protocoles d'authentification plus complexes entre le **Micro serveur** et le **Micro client**.

À partir d'une connexion de rechiffrement établie avec la **Cible** et d'une **Connexion d'importation**, l'**Hôte ECI** peut instancier une session de **Pointeur de média** du mode (mode de rechiffrement, de synchronisation et de format de données) souhaité par l'application et pouvant être pris en charge par le **Micro serveur**.

Une fois la connexion de rechiffrement établie, l'**Hôte ECI** peut instancier une session de **Pointeur de média** avec un **Micro serveur** et commencer à rechiffrer le contenu issu d'une **Connexion d'importation** établie pour la **Cible** (**Client ECI** ou groupe de **Clients ECI**). Plusieurs rechiffrements simultanés du même contenu peuvent être instanciés, chacun au moyen de sa propre session de **Pointeur de média**. Il appartient à l'**Hôte ECI** de veiller à ce que le contenu destiné à la session de **Pointeur de média** visant le rechiffrement provienne du **Pointeur de média** d'exportation authentifié dans la **Connexion d'exportation**. Une mauvaise connexion non autorisée entraînera un échec de l'authentification de l'exportation.

Les mots de contrôle de rechiffrement sont appliqués au contenu déchiffré importé et de nouveaux marqueurs (informations URI, etc.) sont appliqués au contenu rechiffré au moyen du **Système de sécurité évoluée**.

Trois *modes de chiffrement* principaux sont possibles:

1) Mode streaming en ligne: le **Micro serveur** et le **Micro client** sont actifs en même temps. Ils échangent des messages directement (via un canal IP) ou sous la forme de messages explicites par l'intermédiaire de leurs **Hôtes ECI**.

2) Mode streaming hors ligne: le **Micro serveur** chiffre le contenu "à la volée" et émet régulièrement de nouvelles données nécessaires au déchiffrement par le **Micro client**. Le résultat peut être différé (mode de décalage temporel) ou stocké.

3) Mode stockage hors ligne: le **Micro serveur** chiffre le contenu, puis, une fois terminé, produit les données requises par le **Micro client** au début du décodage du contenu.

Le schéma à la Figure 9.7.2.5.1-1 présente une vue d'ensemble des différents modes de chiffrement.

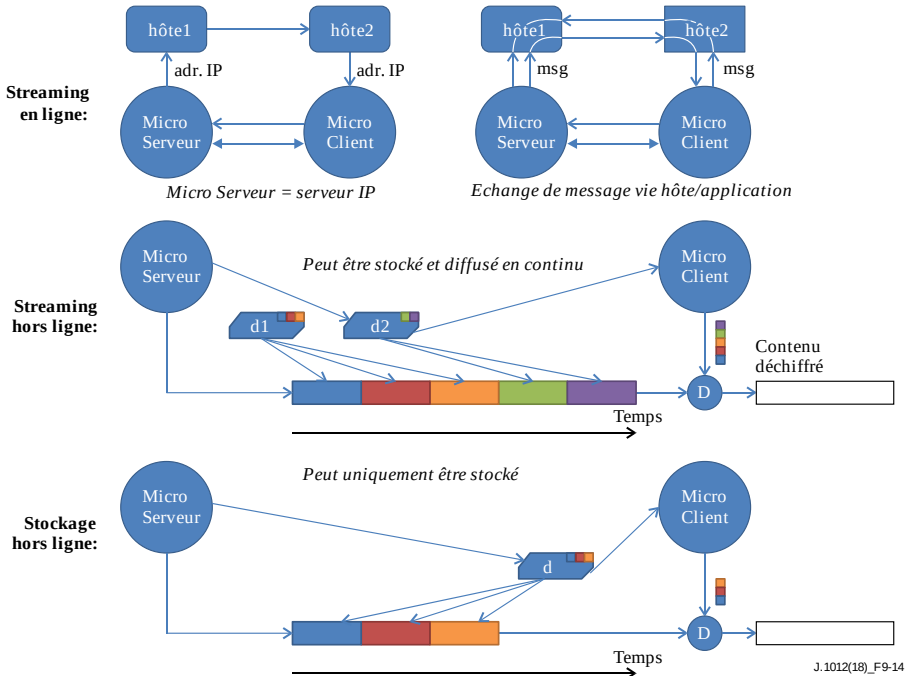


Figure 9.7.2.5.1-1 – Modes de chiffrement des sessions Micro DRM

Les données nécessaires au déchiffrement du contenu devant être échangé dans les deux modes de chiffrement hors ligne entre le **Micro serveur** et le **Micro client** peuvent être transmises selon les *modes de format de données* suivants:

1) Mode générique: le **Micro serveur** produit des conteneurs de données opaques contenant les informations requises par le **Micro client** pour déchiffrer le contenu.

2) Mode ISOBMFF (uniquement si le *mode de synchronisation* est le mode fichier): le **Micro serveur** génère des boîtes PSSH destinées à être incluses dans un fichier ISOBMFF (voir la norme [ISO/CEI 14496-12]). L'**Hôte ECI** peut créer des fichiers ISOBMFF en incluant correctement ces boîtes PSSH dans des boîtes ISOBMFF MOOV ou MOOF.

Deux mécanismes sont pris en charge pour le *mode de synchronisation* en vue d'associer le mot de contrôle valide à une section de contenu, et sont applicables à tous les modes de rechiffrement ci-dessus:

1) En mode flux de transport (bit alterné), le **Micro serveur** produit des sections ECM pouvant être mises en paquets et insérées dans le flux de transport par l'**Hôte ECI**. L'ECM est inséré avant la période cryptographique pour laquelle il fournit des informations afin de permettre le calcul du mot de contrôle.

2) En mode fichier, le **Micro serveur** produit des mots de contrôle chiffrés référencés par des identificateurs de clé explicites dans les informations de déchiffrement supplémentaires. L'**Hôte ECI** doit préserver l'association basée sur l'identificateur de clé entre la section de contenu chiffrée et un mot de contrôle spécifique afin que le **Micro client** puisse produire le mot de contrôle valide pour le désembrouillage.

En mode hors ligne, la synchronisation des données supplémentaires est nécessaire pour le déchiffrement ou le calcul de l'identificateur de clé ou des ECM faisant explicitement référence à la relation de dépendance temporelle entre les données et l'identificateur de clé ou le numéro d'ECM.

Tous les **Micro serveurs** ne doivent pas nécessairement prendre en charge tous les modes de fonctionnement. Lors de l'initialisation, juste après avoir utilisé l'API de découverte, un **Micro serveur** communique les modes (combinaison du mode de chiffrement, du mode de format de données et du mode de synchronisation) qu'il peut prendre en charge.

Une fois la session de **Pointeur de média** instanciée, elle peut être démarrée et arrêtée par l'**Hôte ECI** et annulée par le **Client ECI**.

Les messages de l'API de rechiffrement sont répertoriés dans le Tableau 9.7.2.5.1-1.

Tableau 9.7.2.5.1-1 – Messages de l'API de rechiffrement

Message	Type	Sens	Étiquette	Description
setEncrModes	set	C→H	0x0	Le **Micro serveur** informe l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.
reqEncrTargets	A	H→C	0x1	L'**Hôte ECI** demande au **Micro serveur** de fournir les nœuds de **Cible** qu'il peut authentifier en vue du déchiffrement.
reqEncrConnSetup	A	H→C	0x2	L'**Hôte ECI** demande au **Client ECI** de créer une connexion avec la **Cible** du rechiffrement et de pré-authentifier la **Cible** du rechiffrement pour référence ultérieure lors de la configuration d'une session de **Pointeur de média**.
reqEncrConnDrop	A	H→C	0x3	L'**Hôte ECI** demande au **Client ECI** d'abandonner toute information relative à une connexion de rechiffrement précédemment pré-authentifiée.
reqEncrConnCancel	A	C→H	0x4	Le **Client ECI** annule une connexion établie précédemment avec la **Cible** du chiffrement.
reqEncrMhOpen	A	H→C	0x5	L'**Hôte ECI** demande au **Client ECI** d'ouvrir une session de **Pointeur de média** afin de rechiffrer le contenu issu d'une **Connexion d'importation** entrante à destination d'une connexion de rechiffrement établie.
reqEncrMhClose	A	H→C	0x6	L'**Hôte ECI** ferme la **Session de rechiffrement** avec le **Client ECI**.
reqEncrMhCancel	A	C→H	0x7	Le **Client ECI** met fin à la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.
reqEncrMhStart	A	H→C	0x8	L'**Hôte ECI** demande au **Client ECI** de démarrer l'opération de rechiffrement pour une session de **Pointeur de média**.
reqEncrMhStop	A	H→C	0x9	L'**Hôte ECI** demande au **Client ECI** d'arrêter une opération de rechiffrement pour une session de **Pointeur de média**.
reqEncrMhQuit	A	C→H	0xA	Le **Client ECI** informe l'**Hôte ECI** que l'opération de rechiffrement sur le **Pointeur de média** a pris fin.
reqEncrIpServer	A	H→C	0xB	L'**Hôte ECI** demande l'adresse de serveur IP d'un **Micro serveur** afin de permettre aux **Micro clients** de créer des connexions IP.
reqEncrMsgSend	A	C→H	0xC	Le **Micro serveur** demande à l'**Hôte ECI** de transférer un message à la **Cible** d'une session de **Pointeur de média**.
reqEncrMsgRecv	A	H→C	0xC	L'**Hôte ECI** fournit au **Micro serveur** un message issu de la **Cible** d'une session de **Pointeur de média**.

Tableau 9.7.2.5.1-1 – Messages de l'API de rechiffrement

Message	Type	Sens	Étiquette	Description
reqEncrTsData	A	C→H	0xE	Le **Micro serveur** fournit à l'**Hôte ECI** les données devant être transférées au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'ECM.
reqEncrTsEcm	A	C→H	0xF	Le **Micro serveur** émet une section ECM requise par le **Micro client** pour le déchiffrement au cours de la période cryptographique suivante.
reqEncrFileData	A	C→H	0x10	Le **Micro serveur** fournit à l'**Hôte ECI** un message devant être transféré au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'identificateur de clé.

9.7.2.5.2 Message setEncrModes

C→H setEncrModes(EciEncrModes **modes)**

- Ce message permet au **Micro serveur** d'informer l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.

Définitions des paramètres de la requête:

modes: EciEncrModes	Modes de chiffrement pris en charge par le **Micro serveur**. Le type EciEncrModes est spécifié dans le Tableau 9.7.2.5.2-1.

Tableau 9.7.2.5.2-1 – Définition du type EciEncrModes

```
Typedef uint EciEncrModes;
```

Définitions des bits:

Nom	Bit	Mode pris en charge par le Micro serveur lorsque la valeur est égale à 0b1
OnlineIpMode	0	Le mode IP en ligne est pris en charge.
OnlineMsgMode	1	Le mode message en ligne est pris en charge.
OfflineStreamMode	2	Le mode streaming hors ligne est pris en charge.
OfflineStorageMode	3	Le mode stockage hors ligne est pris en charge.
OfflineDataMode	4	Les conteneurs de format de données par défaut sont pris en charge pour les données de déchiffrement en mode hors ligne. Non pertinent si aucun mode hors ligne n'est sélectionné.
OfflineIsobmffMode	5	Les boîtes PSSH du format ISOBMFF sont prises en charge pour les données de déchiffrement en mode hors ligne. Non pertinent si aucun mode hors ligne n'est sélectionné.
SyncTs	6	Synchronise les mots de contrôle avec les périodes cryptographiques délimitées par le bit alterné du format flux de transport.
SyncFile	7	Synchronise au format de type fichier au moyen d'identificateurs de clé afin d'associer les sections de contenu à leur mot de contrôle.
autre	RFU	Réservé à une utilisation future.

9.7.2.5.3 Message reqEncrTargets

H→C reqEncrTargets() →
C→H resEncrTargets(EncrTarget **target[])**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** de fournir les cibles de chiffrement qu'il peut authentifier.

Définitions des paramètres de la réponse:

target: EncrTarget[]	Liste de cibles de chiffrement que le **Micro serveur** peut authentifier. La définition du type EncrTarget est spécifiée dans le Tableau 9.7.2.5.3-1.

Tableau 9.7.2.5.3-1 – Définition du type EncrTarget

```
typedef struct EncrTarget {
   uint   targetType;
   byte   target[8];
} EncrTarget;
```

Définitions des champs:

targetType: uint	Type de cible de chiffrement: client individuel si la valeur est égale à 1, groupe de clients si la valeur est égale à 2; les autres valeurs sont réservées à une utilisation future.
target: byte[8]	Identificateur représentant la cible. La valeur est définie dans les limites du **Système Micro DRM**. La mise en correspondance par l'**Hôte ECI** est définie en termes d'égalité entre les champs **targetType** et **target**.

Sémantique détaillée:

- L'**Hôte ECI** peut mettre en correspondance des **Micro clients cibles** sur la base de la **Cible**. Ce sont l'application et/ou l'**Hôte ECI** qui localisent les **Micro clients** candidats potentiels.
- Les **Hôtes ECI** souhaitant utiliser des fonctionnalités d'enregistreur vidéo personnel et de visionnement différé en local (en utilisant un support de stockage intégré ou connecté/en réseau sur lequel ils peuvent stocker le contenu chiffré et les données associées) peuvent essayer de mettre en correspondance un **Micro serveur** capable de fonctionner en mode **OfflineStreamMode** avec des **Micro clients** installés sur le même **Hôte ECI**.

9.7.2.5.4 Message reqEncrConnSetup

H→C reqEncrConnSetup(ushort **targetConnId**, EciEncrTarget **target**, ushort **credLen**, byte **cred[])**
C→H resEncrConnSetup(ushort **targetConnId)**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** de créer une connexion de rechiffrement avec la **Cible** et de (pré-)authentifier cette dernière. Les codes d'erreur sont définis dans le Tableau 9.7.2.5.19-1.

Définitions des paramètres de la requête:

targetConnId: ushort	Identificateur utilisé ultérieurement comme référence de la **Cible** par l'**Hôte ECI** et le **Micro serveur**.
target: EciEncrTarget	Identificateur représentant la **Cible** en vue de l'authentification. La valeur est définie dans les limites du **Système Micro DRM**. La mise en correspondance par l'**Hôte ECI** est définie en termes d'égalité entre les champs **targetType** et **target**.
credLen: ushort	Longueur du paramètre cred en octets.
cred: byte[]	Informations relatives aux justificatifs d'identité issues de la **Cible** devant être authentifiée par le **Micro serveur**.

Définitions des paramètres de la réponse:

targetConnId: ushort	Identificateur utilisé ultérieurement comme référence de la **Cible** par l'**Hôte ECI** et le **Micro serveur**.

Sémantique détaillée:

- Si **targetConnId** est égal à un paramètre **targetConnId** utilisé précédemment par l'**Hôte ECI** mais n'ayant pas été abandonné par la suite, la **Cible** précédente associée à **targetConnId** est remplacée ou mise à jour.

Préconditions de la requête:

1) La **Cible** doit être identique à une **Cible** fournie précédemment à l'**Hôte ECI** par le **Micro serveur** dans un message **resEncrTargets**. Dans le cas contraire, une erreur est renvoyée pour ce paramètre.

2) La **Cible** doit correspondre à une **Cible** fournie par le **Micro client** et permettre l'authentification au moyen du paramètre **cred**.

Postconditions de la réponse:

1) Le statut d'authentification est renvoyé. Il convient de noter que le résultat ne sera pas nécessairement concluant et pourra par exemple fournir les mauvais justificatifs d'identité, rendant impossible le décodage du contenu chiffré.

2) L'**Hôte ECI** peut utiliser le paramètre **targetConnId** pour faire référence à la **Cible** (pré-)authentifiée.

Tableau 9.7.2.5.4-1 – Codes d'erreur du message reqEncrConnSetup

Nom	Description
ErrEncrAuthFail	Voir le Tableau 9.7.2.5.19-1.
ErrEncrAuthInconclusive	

9.7.2.5.5 Message reqEncrConnDrop

H→C reqEncrConnDrop(ushort **targetConnId) →**
C→H resEncrConnDrop(ushort **targetConnId)**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** d'abandonner toute information relative à une connexion de rechiffrement précédemment (pré-)authentifiée.

Définitions des paramètres de la requête:

targetConnId: ushort	Identificateur de la connexion avec la **Cible** devant être supprimée par le **Micro serveur**.

Définitions des paramètres de la réponse:

targetConnId: ushort	Identificateur de la connexion avec la **Cible** supprimée du **Micro serveur**.

Préconditions de la requête:

1) L'identificateur **targetConnId** devrait exister sur le **Micro serveur**.

Préconditions de la réponse:

1) Le **Micro serveur** n'associe plus **targetConnId** à une connexion pré-authentifiée avec une **Cible** et a libéré toutes les ressources associées à la pré-authentification de **targetConnId**.

9.7.2.5.6 Message reqEncrConnCancel

C→H reqEncrConnCancel(ushort **targetConnId) →**
H→C resEncrConnDrop(ushort **targetConnId)**

- Ce message permet au **Micro serveur** d'informer l'**Hôte ECI** qu'il a abandonné toute information relative à une connexion de rechiffrement précédemment (pré-)authentifiée.

Définitions des paramètres de la requête:

targetConnId: ushort	Identificateur de la connexion avec la **Cible** annulée par le **Micro serveur**.

Définitions des paramètres de la réponse:

targetConnId: ushort	Identificateur de la connexion avec la **Cible** annulée par le **Micro serveur**.

Préconditions de la requête:

1) L'identificateur **targetConnId** devrait exister sur le **Micro serveur**.

Préconditions de la réponse:

1) La valeur de **TargetConnId** a été libérée et pourra être attribuée à nouveau par l'**Hôte ECI** dans un message reqEcnrConnSetup ultérieur.

9.7.2.2.7 Message reqEncrMhOpen

H→C reqEncrMhOpen(ushort **mh,** ushort **impConn,** ushort **targetConnId,** EncrMode **mode) → C→H resEncrMhOpen(**ushort **mh)**

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** d'ouvrir une session de Pointeur de média afin de rechiffrer le contenu issu d'une **Connexion d'importation** entrante, sous le contrôle du **Micro serveur**, pour le transférer vers une **Cible** pré-authentifiée. Les codes d'erreur sont définis dans le Tableau 9.7.2.5.7-1.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement à ouvrir, attribué par l'**Hôte ECI**.
impConn: ushort	Identificateur de la connexion d'entrée à partir de laquelle le contenu doit être rechiffré.
targetConnId: ushort	Identificateur de la connexion avec la **Cible** pour laquelle le contenu doit être rechiffré.
mode: EncrMode	Spécification du mode unique (de chiffrement, de format de données, de synchronisation) devant être utilisé par le **Micro serveur**, sélectionné parmi les modes pris en charge par le **Micro serveur** tels qu'indiqués par le message **setEncrModes**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement à ouvrir, attribué par l'**Hôte ECI**.

Préconditions de la requête:

1) L'**Hôte ECI** a réservé toutes les ressources requises pour la session à créer.

2) **impConn** et **targetConnId** sont établis par l'**Hôte ECI** avec le **Micro serveur**.

Préconditions de la réponse:

1) En cas de réussite, le **Micro serveur** a réservé toutes les ressources généralement nécessaires pour rechiffrer du contenu pour la session demandée, y compris l'accès à toutes les ressources externes (serveurs DRM, **Cartes à puce**, etc.) requises en principe pour une opération de déchiffrement.

NOTE – Les ressources nécessaires à titre exceptionnel ou les ressources pouvant normalement être obtenues à la demande sont exclues.

2) Si l'erreur **ErrEncrUserDelay** est renvoyée, le **Micro serveur** attend une action de la part de l'**Utilisateur** pour pouvoir ouvrir la session (par exemple pour accéder à une **Carte à puce** ou obtenir une authentification de l'**Utilisateur**). L'**Hôte ECI** peut continuer d'envoyer la **requête reqEncrMhOpen** (avec les mêmes paramètres) jusqu'à ce qu'un résultat positif ou une erreur définitive soit renvoyé(e), ou peut envoyer une requête **reqEncrMhClose** pour terminer la session en cours. Le **Micro serveur** peut annuler la demande au moyen du message **reqEncrMhCancel** s'il ne parvient pas à obtenir la réponse requise de la part de l'**Utilisateur**.

Tableau 9.7.2.5.7-1 – Codes d'erreur du message reqEncrMhOpen

Nom	Description
ErrEncrUserMissing	Voir le Tableau 9.7.2.5.19-1.
ErrEncrCardMissing	
EnrEncrServiceMissing	
ErrEncrResourceMissing	
ErrEncrMmiMissing	
ErrEncrClientAuthError	

9.7.2.5.8 Message reqEncrMhClose

H→C reqEncrMhClose(ushort **mh) →**
C→H resEncrMhClose(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de fermer une **Session de rechiffrement** avec le **Micro serveur**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement à fermer.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement à fermer.

Préconditions de la requête:

1) La session de **Pointeur de média** est à l'état ouvert (ou une erreur se produira).

Préconditions de la réponse:

1) Les ressources requises par le **Micro serveur** pour maintenir la session sont libérées.
2) **mh** est à l'état fermé par le client.

9.7.2.5.9 Message reqEncrMhCancel

C→H reqEncrMhCancel(ushort **mh,** uchar **reason) →**
H→C resEncrMhCancel(ushort **mh)**

- Ce message permet au **Client ECI** de fermer une **Session de rechiffrement** avec le **Client ECI** exportateur (**Micro serveur**) spécifié.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement annulée par le **Micro serveur**.
reason: uchar	Raisons de l'annulation de la session de rechiffrement. Les valeurs sont définies dans le Tableau 9.7.2.5.9–1.

Tableau 9.7.2.5.9-1 – Valeurs des raisons du message reqEncrMhCancel

Nom	Valeur	Description
EncrMhUndefined	0x00	Une erreur inconnue s'est produite sur le **Micro serveur** et l'a obligé à fermer la session.
EncrMhCardMissing	0x01	Une **Carte à puce** est nécessaire au rechiffrement mais n'a pu être (re-)connectée pour permettre le rechiffrement du contenu dans un délai raisonnable.
EncrMhServiceMissing	0x02	Un service (externe à l'**Équipement CPE**) aidant le **Micro serveur** à fournir les services de chiffrement nécessaires au maintien d'une session de déchiffrement n'est pas disponible dans un délai raisonnable.
EncrMhResourceMissing	0x03	Une ressource (interne à l'**Équipement CPE**) nécessaire pour fournir les services de rechiffrement n'est pas disponible pour le **Micro serveur** dans un délai raisonnable (à l'exclusion de **DcrMhMmiMissing**).
EncrMhMmiMissing	0x04	Le **Micro serveur** n'est pas parvenu à atteindre une ressource de session d'interface homme-machine destinée à l'interaction avec l'**Utilisateur** et requise pour maintenir la **Session de rechiffrement** dans un délai raisonnable.
RFU	Autre	Réservé à une utilisation future.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement annulée.

Préconditions de la requête:

1) Le **Client ECI** a libéré toutes les ressources dont il a eu besoin en particulier pour la session.

Postconditions de la requête:

1) L'**Hôte ECI** peut libérer toute ressource liée au **Pointeur de média**.

Postconditions de la réponse:

1) L'**Hôte ECI** ferme la session de **Pointeur de média**.

9.7.2.5.10 Message reqEncrMhStart

H→C reqEncrMhStart(ushort **mh) →**
C→H resEncrMhStart(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** de démarrer l'opération de rechiffrement pour une session de **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement à démarrer.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement démarrée.

Préconditions de la requête:

1) La session de **Pointeur de média** est à l'état ouvert (ou une erreur se produira).

Préconditions de la réponse:

1) La session de **Pointeur de média** a démarré (ou une erreur s'est produite).

Sémantique détaillée:

- Le chiffrement du contenu se fait à mesure qu'il est fourni par le **Client ECI** exportateur.
- En cas de conflit entre les informations URI ou si le **Client ECI** exportateur ne parvient pas à authentifier le **Micro serveur** vers lequel exporter le contenu, aucun contenu chiffré ne sera produit, le statut des informations URI de contrôle de sortie du **Micro serveur** sera défini sur OcAnyOther avec la valeur 0b1, tous les autres bits de contrôle des données de sortie seront

définis sur 0b0 (signifiant qu'aucune sortie n'est autorisée). Le **Micro serveur** continuera à tenter de chiffrer le contenu lorsqu'il y est autorisé.

- Les messages d'initialisation à destination du **Micro client** sont mis à disposition par l'intermédiaire des messages respectifs à cet effet. Pour les sessions dont le mode de rechiffrement est **OfflineStreamMode**, les premières données d'initialisation destinées au déchiffrement du contenu sont produites peu après le message **resEncrMhStart**.
- L'envoi d'un deuxième message **reqEncrMhStart** avant la fin du processus de chiffrement mettra fin à ce dernier et démarrera le suivant.

9.7.2.5.11 Message reqEncrMhStop

H→C reqEncrMhStop(ushort **mh) →**
C→H resEncrMhStop(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** d'arrêter l'opération de rechiffrement pour une session de **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement à arrêter.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement arrêtée.

Préconditions de la requête:

1) La session de **Pointeur de média** est à l'état démarré (ou une erreur se produira).

Préconditions de la réponse:

1) La session de **Pointeur de média** a pris fin.

Postconditions de la réponse:

1) L'**Hôte ECI** peut réutiliser la valeur de la session de **Pointeur de média**.

Sémantique détaillée:

- Pour les sessions dont le mode de chiffrement est **OfflineStorageMode**, les données de déchiffrement finales sont produites avant que le **Micro serveur** n'envoie le message **resEncrMhStop**. Ceci est valable également pour toute donnée de déchiffrement finale éventuellement nécessaire au déchiffrement dans d'autres types de sessions.

9.7.2.5.12 Message reqEncrMhQuit

C→H reqEncrMhQuit(ushort **mh,** uchar **reason) →**
C→H resEncrMhQuit(ushort **mh)**

- Ce message permet au **Micro serveur** d'informer l'**Hôte ECI** que l'opération de rechiffrement associée à la session de **Pointeur de média** a pris fin.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement terminée.
reason: uchar	Raisons telles qu'indiquées dans le Tableau 9.7.2.5.9–1.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement terminée.

Préconditions de la requête:

1) La session de **Pointeur de média** était à l'état démarré mais est maintenant arrêtée.

Préconditions de la réponse:

1) L'**Hôte ECI** sait que le chiffrement de la session est à l'état non démarré.

Sémantique détaillée:

- Si l'erreur est de nature quasi permanente, le **Micro serveur** peut également annuler la session de **Pointeur de média** proprement dit.
- Si le **Micro serveur** peut produire des données de déchiffrement valides avant de mettre fin à la **Session de rechiffrement**, lorsqu'il s'agit d'une session en mode de chiffrement **OfflineStorageMode**, les données de déchiffrement finales sont produites avant que le **Micro serveur** n'envoie le message **resEncrMhQuit**. Ceci est valable également pour toute donnée de déchiffrement finale éventuellement nécessaire au déchiffrement dans d'autres types de sessions.

9.7.2.5.13 Message reqEncrIpServer

H→C reqEncrIpServer(ushort **mh) →**
C→H resEncrIpServer(ushort **mh**, Addrinfo **addr)**

- Ce message permet à l'**Hôte ECI** de demander au **Micro serveur** de fournir l'adresse IP de la **Cible** pour les connexions IP entrantes issues des **Micro clients**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle une adresse IP destinée aux messages ou aux connexions entrants est requise.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle une adresse IP destinée aux messages ou aux connexions entrants est requise.
addr: Addrinfo	Protocole/adresse/port IP pour les messages ou les connexions entrants issus d'un **Micro client**.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineIpMode**.

Préconditions de la réponse:

1) L'**Hôte ECI** sait que le chiffrement de la session est à l'état non démarré.

Sémantique détaillée:

- L'échange d'adresse IP entre le **Micro client** et le **Micro serveur** est spécifique au **Système Micro DRM**. Cela inclut le choix du protocole et toute convention relative à la fin d'une connexion ou d'un échange lors d'une session de diffusion de contenu.
- Ce message peut-être émis lors d'une session de **Pointeur de média** dont le processus de rechiffrement n'a pas encore été démarré.

Tableau 9.7.2.5.13-1 – Codes d'erreur du message reqEncrIpServer

Nom	Description
ErrEncrIpNone	Voir le Tableau 9.7.2.5.19-1.

9.7.2.5.14 Message reqEncrMsgSend

C→H reqEncrMsgSend(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resEncrMsgSend(ushort **mh)**

- Ce message permet au **Micro serveur** de demander à l'**Hôte ECI** de transférer un message au **Micro client** ou **Micro clients cibles** (en présence d'un groupe cible) associés au **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle un message doit être transféré au **Micro client cible**.
length: uint	Longueur du champ **msg** en octets.
msg[]: byte	Message devant être transféré au **Micro client**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineMsgMode**.

Préconditions de la réponse:

1) Le message a été transféré au **Micro client**; l'**Hôte ECI** est prêt à accepter un nouveau message **reqEncrMsgSend**.

Sémantique détaillée:

- L'**Hôte ECI** doit être capable de traiter et de transférer au moins un message à la fois au **Micro client**. Les messages doivent être envoyés dans l'ordre. L'**Hôte ECI** n'est pas obligé de fournir une mise en mémoire tampon spécifique pour plusieurs requêtes **reqEncrMsgSend** simultanées en attente. Une implémentation de **Micro serveur** sécurisée doit utiliser le message **resEncrMsgSend** en tant qu'établissement d'une liaison de flux de contrôle.
- Le mécanisme de transfert de l'**Hôte ECI** doit être suffisamment fiable pour éviter un échec des applications normales (perte de message ou un message sur 10 000 dans le mauvais ordre). En ce qui concerne les applications pour lesquelles il existe un risque que des informations d'accès essentielles (relatives au contenu chiffré) soient perdues de manière définitive ou qu'un visionnement de qualité soit altéré, il est recommandé de prendre des précautions supplémentaires au niveau application.

9.7.2.5.15 Message reqEncrMsgRecv

H→C reqEncrMsgRecv(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resEncrMsgRecv(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de fournir au **Micro serveur** un message provenant du **Micro client cible**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle le **Micro serveur** reçoit un message de la part du **Micro client cible**.
length: uint	Longueur du champ **msg** en octets.
msg: byte[]	Message devant être reçu par le **Micro serveur**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle une adresse IP destinée aux messages ou aux connexions entrants est requise.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineMsgMode**.

Préconditions de la réponse:

1) Le message a été traité par le **Micro serveur**, qui est prêt à accepter un nouveau message **reqEncrMsgRecv**.

Sémantique détaillée:

- Le **Micro serveur** doit traiter au moins un message à la fois. Le **Micro serveur** n'est pas obligé de fournir une mise en mémoire tampon spécifique pour plusieurs requêtes **reqEncrMsgSend** simultanées en attente; il doit toutefois prendre garde au fait qu'il est alors prêt à traiter le message suivant en raison des autres exigences de réactivité le concernant. Une implémentation d'**Hôte ECI** sécurisée doit utiliser le message **resEncrMsgRecv** en tant qu'établissement d'une liaison de flux de contrôle.
- La fiabilité du service de transfert entre le **Micro client** et le **Micro serveur** est telle que définie pour le message **reqEncrMsgSend** au § 9.7.2.5.14.

9.7.2.5.16 Message reqEncrTsData

C→H reqEncrTsData(ushort **mh,** TsSync **sync,** uint **length,** byte **msg[]) →**
C→H resEncrTsData(ushort **mh)**

- Ce message permet au **Micro serveur** de fournir à l'**Hôte ECI** les données devant être transférées au **Micro client cible** d'un **Pointeur de média** en vue de déchiffrer le contenu, y compris les informations de synchronisation liées à l'ECM.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement.
sync: TsSync	Synchronisation des informations liées à un identificateur ecmId associé au contenu. Le Tableau 9.7.2.5.16-1 fournit des détails complémentaires.
length: uint	Longueur en octets du message à transférer.
msg: byte[]	Message devant être transféré au **Micro client**.

Tableau 9.7.2.5.16-1 – Définition du type TsSync

```
typedef struct TsSync {
   uint  ecmId;
   uint  precTime;
} TsSync;
```

Définitions des champs:

ecmId: uint	Numéro d'identification d'un ECM associé au contenu devant être précédé par ce message de données destiné au **Micro client**.
precTime: uint	Temps réel de lecture du contenu en unités de 100 ms, pour un maximum de 300 secondes, pendant lequel ce message doit précéder l'application d'un ECM ecmId au processus de décodage du contenu.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement pour laquelle une adresse IP destinée aux messages ou aux connexions entrants est requise.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte, en *mode de rechiffrement* **OfflineStream** ou **OfflineStorage**; elle utilise le *mode de format de données* **OfflineDataMode** et le *mode de synchronisation* **SyncTs**.

Préconditions de la réponse:

1) L'**Hôte ECI** est prêt à recevoir le message de données suivant.

Sémantique détaillée:

- L'**Hôte ECI** doit veiller à ce que le **Micro client** reçoive les données conformément aux exigences de synchronisation, parallèlement au contenu chiffré.

- L'**Hôte ECI** doit mettre les données du message en mémoire tampon de manière appropriée (en tant que données associées au contenu) et doit répondre au message suivant dans les délais tels que proposés dans le document [b-UIT-T J Suppl. 7].
- Lors d'un fonctionnement en mode **OfflineStream**, il est possible que le **Micro serveur** produise un ou plusieurs messages de données avant une **Session de rechiffrement** démarrée.
- En mode **OfflineStorage**, le **Micro serveur** produira au maximum un message de données à la fin de la session de chiffrement. Ce message peut être précédé de l'ECM avec lequel il est censé se synchroniser. D'où le mode "stockage hors ligne". De manière générale, ce message de données doit être traité par le **Micro client** avant le contenu et les ECM.

9.7.2.5.17 Message reqEncrTsEcm

C→H reqEncrTsEcm(ushort **mh,** uint **ecmId,** uint **length,** byte **ecm[]) →**
C→H resEncrTsEcm(ushort **mh)**

- Ce message permet au **Micro serveur** d'émettre une section ECM requise pour le déchiffrement au cours de la période cryptographique suivante.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement.
ecmId: uint	Numéro d'identification de l'ECM attribué par le **Micro serveur** dans le but de synchroniser les messages de données.
length: uint	Longueur du paramètre **ecm** en octets; le format d'**ecm** présente une seule section.
ecm: byte[]	Message ECM devant être inséré lors de la période cryptographique suivante.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte et utilise le *mode de synchronisation* **SyncTs**.

Préconditions de la réponse:

1) L'**Hôte ECI** est prêt à insérer l'ECM suivant.

Sémantique détaillée:

- L'**Hôte ECI** insérera l'ECM au sein du flux de transport dans un certain délai après réception du message. Des valeurs de délai sont proposées dans le document [b-UIT-T J Suppl. 7]. L'ECM sera répété à un intervalle raisonnable (défini dans la norme [ISO/CEI 13818-1]). Le PID de l'ECM sera libre et généré par l'**Hôte ECI**.
- L'**Hôte ECI** peut mettre à jour les informations relatives à la PMT dans le flux, pouvant refléter le PID de l'ECM, ou transférera les informations relatives au PID de l'ECM afin de permettre à un **Micro client** de récupérer à un stade ultérieur les informations de déchiffrement requises.
- Lors du changement d'un élément de contenu et/ou d'un chiffrement de couche supérieure, le **Micro serveur** peut émettre deux messages ECM successifs différents pour la même période cryptographique à venir. L'**Hôte ECI** insérera au minimum le dernier pour le reste de la période. En mode visionnement différé/stockage, il insérera le dernier ECM pour l'ensemble de la période cryptographique.

9.7.2.5.18 Message reqEncrFileData

H→C reqEncrFileData(ushort **mh,** byte **syncKid[**MaxUuidLen**],** uint **datalength,** byte **data[])**
C→H resEncrFileData(ushort **mh)**

- Ce message permet au **Micro serveur** de fournir à l'**Hôte ECI** un message devant être transféré au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'identificateur de clé.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de chiffrement.
syncKid [MaxUuidLen]**:** byte	Identificateur de clé qui sera utilisé pour chiffrer le prochain "fragment" du fichier pour lequel le **Micro client** a besoin des données associées en vue du déchiffrement.
datalength: uint	Longueur des données en octets.
data[]**:** byte	Données destinées au **Micro client** à des fins de déchiffrement. Le format des données est opaque si le mode de format des données est **OfflineDataMode** et consiste en une boîte PSSH devant être incluse dans une boîte MOOV ou MOOF ISOBMFF si le mode de format des données est **OfflineIsobmffMode**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte, en *mode de rechiffrement* **OfflineStream** ou **OfflineStorage** et en *mode de synchronisation* **SyncFile**.

Préconditions de la réponse:

1) L'**Hôte ECI** est prêt à recevoir le message de données suivant.

Sémantique détaillée:

- L'**Hôte ECI** doit veiller à ce que le **Micro client cible** reçoive les données conformément aux exigences de synchronisation parallèlement au contenu chiffré.
- L'**Hôte ECI** créera un fichier ISOBMFF valide incluant la boîte PSSH fournie, ou veillera à ce que les données soient transmises avec le contenu du fichier au **Micro client** et fournies au **Micro client** conformément aux exigences de synchronisation des données.
- L'**Hôte ECI** doit mettre les données du message **reqEncrMsgRecv** en mémoire tampon de manière appropriée (en tant que données associées au contenu). Des valeur pour les exigences en termes de temps de réponse sont proposées dans le document [b-UIT-T J Suppl. 7].
- Lors d'un fonctionnement en mode **OfflineStream**, il est possible que le **Micro serveur** produise un ou plusieurs messages de données avant une **Session de rechiffrement** démarrée.
- En mode **OfflineStorage**, le **Micro serveur** produira au maximum un message de données à la fin de la session de chiffrement. De manière générale, ce message de données doit être traité par le **Micro client** avant tout contenu.

9.7.2.5.19 Codes d'erreur de l'API de rechiffrement

Tableau 9.7.2.5.19-1 – Codes d'erreur de l'API de rechiffrement

Nom	Valeur	Description
ErrEncrAuthInconclusive	1	L'authentification n'a été réalisée que partiellement et n'a pas été concluante, mais aucune erreur ne s'est produite.
ErrEncrAuthFail	-256	Il n'a pas été possible d'identifier le statut de l'authentification parentale de l'élément de contenu, mais l'authentification parentale a été réalisée et s'est avérée correcte.
ErrEncrUserMissing	-257	L'**Utilisateur** n'a pas fourni une donnée essentielle au **Micro serveur** pour effectuer ou continuer le rechiffrement du contenu.
ErrEncrCardMissing	-258	Une **Carte à puce** est nécessaire au rechiffrement mais n'a pu être (re-)connectée pour permettre le rechiffrement du contenu dans un délai raisonnable.
EnrEncrServiceMissing	-259	Un service (externe à l'**Équipement CPE**) aidant le **Micro serveur** dans une session de déchiffrement n'est pas disponible dans un délai raisonnable.
ErrEncrResourceMissing	-260	Une ressource non spécifiée de l'**Équipement CPE** nécessaire au traitement et/ou au rechiffrement du contenu n'est pas disponible.
ErrEncrMmiMissing	-261	L'accès du **Micro serveur** à l'interface homme-machine est requis mais n'est pas disponible.
ErrEncrClientAuthError	-262	Le **Micro serveur** ne parvient pas à authentifier le **Micro client cible**.
ErrEncrIpNone	-263	Le **Micro serveur** ne peut pas fournir d'adresse IP pour la communication avec le **Micro client**.

9.7.2.6 API de redéchiffrement par le Micro client

9.7.2.6.1 Généralités

L'API de déchiffrement par le **Micro client** permet à un **Micro client** de redéchiffrer du contenu d'un **Micro serveur**.

La phase de découverte permet au **Micro client** de communiquer les cibles de déchiffrement pour lesquelles il peut proposer des services de déchiffrement et fournir les justificatifs d'identité avec lesquels un **Micro serveur** peut créer une connexion authentifiée avec ce même **Micro client** constituant alors une **Cible**.

Le **Micro client** doit prendre en charge les modes de déchiffrement correspondant aux modes de chiffrement proposés par son **Micro serveur** complémentaire. Le **Micro client** peut déchiffrer les services au moyen d'un des modes couramment pris en charge, sur la base de l'API de déchiffrement commune.

Cette API comporte des messages supplémentaires pour transmettre, entre le **Micro serveur** et le **Micro client**, les données destinées au déchiffrement requis, et ce pour les différents modes.

Les messages de l'API de déchiffrement par le **Micro client** sont répertoriés dans le Tableau 9.7.2.6.1-1.

Tableau 9.7.2.6.1-1 – Messages de l'API de déchiffrement

Message	Type	Sens	Étiquette	Description
setDcrModes	set	C→H	0x0	Le **Micro client** informe l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.
reqDcrTargets	A	H→C	0x1	L'**Hôte ECI** demande au **Micro client** de fournir les cibles de chiffrement pour lesquelles il peut déchiffrer des services.
reqDcrTargetCred	A	H→C	0x2	L'**Hôte ECI** demande au **Client ECI** de fournir les données d'initialisation d'une connexion avec le **Micro serveur** généralement utilisée pour l'authentification de la **Cible**.
reqDcrIpServer	A	C→H	0xA	Le **Micro client** demande à l'**Hôte ECI** de fournir l'adresse IP du **Micro serveur** en vue des communications ultérieures liées à la session de **Pointeur de média**.
reqDcrMsgSend	A	C→H	0xB	Le **Micro client** demande à l'**Hôte ECI** d'envoyer un message au **Micro serveur** d'une session de **Pointeur de média**.
reqDcrMsgRecv	A	H→C	0xC	L'**Hôte ECI** fournit au **Micro client** un message issu du **Micro serveur** d'une session de **Pointeur de média**.
reqDcrTsData	A	C→H	0xD	Le **Micro serveur** fournit à l'**Hôte ECI** les données devant être transférées au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'ECM.
reqDecrFileData	A	C→H	0xF0	Le **Micro serveur** fournit à l'**Hôte ECI** un message devant être transféré au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'identificateur de clé.

9.7.2.6.2 Message setDcrModes

C→H setDcrModes(EciEncrModes **modes)**

- Ce message permet au **Micro client** d'informer l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.

Définitions des paramètres de la requête:

modes: EciEncrModes	Modes de déchiffrement pris en charge par le **Micro client**. Le type EciEncrModes est spécifié dans le Tableau 9.7.1.5.2–1.

9.7.2.6.3 Message reqDcrTargets

H→C reqDcrTargets() →
C→H resDcrTargets(EncrTarget **target[])**

- Ce message permet à l'**Hôte ECI** de demander au **Micro client** de fournir les cibles de chiffrement pour lesquelles il peut procéder au déchiffrement.

Définitions des paramètres de la réponse:

target[]**:** EncrTarget	Liste de cibles de chiffrement que le **Micro serveur** peut authentifier. La définition du type TargetClient est spécifiée dans le Tableau 9.7.2.5.2–1.

Sémantique détaillée:

- L'**Hôte ECI** peut mettre en correspondance des **Micro clients cibles** sur la base de la **Cible**. Ce sont l'application et/ou l'**Hôte ECI** qui localisent les **Micro clients** candidats potentiels.

9.7.2.6.4 Message reqDcrTargetsCred

H→C reqDcrTargetsCred(EncrTarget **target) →**
C→H reqDcrTargetsCred(uint **credLen**, byte **cred[]**)

- Ce message permet à l'**Hôte ECI** de demander au **Micro Client** de fournir les justificatifs d'identité en vue du chiffrement par le **Micro serveur**.

Définitions des paramètres de la requête:

target: EncrTarget[]	**Cible** de chiffrement pour laquelle le **Micro client** doit fournir les justificatifs d'identité en vue du chiffrement du contenu par un **Micro serveur**.

Définitions des paramètres de la réponse:

credLen: uint	Longueur du paramètre cred en nombre d'octets.
cred[]: byte	Justificatifs d'identité dans un format spécifique au **Micro serveur** qui chiffrera le contenu devant être déchiffré par le **Micro client**.

Sémantique détaillée:

- Ce message permet à l'**Hôte ECI** de demander à un **Micro client** de fournir des justificatifs d'identité correspondant au paramètre **target** afin qu'un **Micro serveur** reconnaissant ce dernier puisse chiffrer du contenu à destination du **Micro client**.

9.7.2.6.5 Message reqDcrIpServer

C→H reqDcrIpServer(ushort **mh)** →
C→H resDcrIpServer(ushort **mh,** Addrinfo **addr)**

- Ce message permet au **Micro client** de demander à l'**Hôte ECI** de fournir l'adresse IP du **Micro serveur** en vue des communications ultérieures liées à la session de **Pointeur de média**. Les codes d'erreur associés sont définis dans le Tableau 9.7.2.6.5-1.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de déchiffrement pour laquelle l'adresse IP du **Micro serveur** destinée à l'envoi/à la réception de messages est demandée.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de déchiffrement pour laquelle l'adresse IP du **Micro serveur** destinée à l'envoi/à la réception de messages est fournie.
addr: Addrinfo	Protocole/adresse/port IP du **Micro serveur** pour ce **Pointeur de média**.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineIpMode**.

Préconditions de la réponse:

1) L'**Hôte ECI** sait que le chiffrement de la session est à l'état non démarré.

Sémantique détaillée:

- L'échange d'adresse IP entre le **Micro client** et le **Micro serveur** est spécifique au **Système Micro DRM**. Cela inclut le choix du protocole et toute convention relative à la fin d'une connexion ou d'un échange lors d'une session de diffusion de contenu.
- Ce message peut-être émis lors d'une session de **Pointeur de média** dont le processus de rechiffrement n'a pas encore été démarré.

Tableau 9.7.2.6.5-1 – Codes d'erreur du message reqDcrIpServer

Nom	Description
ErrDcrIpNone	Voir le Tableau 9.7.2.6.10-1.

9.7.2.6.6 Message reqDcrMsgSend

C→H reqDcrMsgSend(ushort **mh,** uint **length,** byte **msg[])** →
C→H resDcrMsgSend(ushort **mh)**

- Ce message permet au **Micro client** de demander à l'**Hôte ECI** de transférer un message au **Micro serveur** de la **Cible** associé au **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de déchiffrement pour laquelle un message doit être transféré au **Micro serveur**.
length: uint	Longueur du champ **msg** en octets.
msg[]: byte	Message devant être transféré au **Micro serveur**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineMsgMode**.

Préconditions de la réponse:

1) Le message a été transféré au **Micro serveur**; l'**Hôte ECI** est prêt à accepter un nouveau message **reqDcrMsgSend**.

Sémantique détaillée:

- L'**Hôte ECI** doit être capable de traiter et de transférer au moins un message à la fois au **Micro serveur**. Les messages doivent être envoyés dans l'ordre. L'**Hôte ECI** n'est pas obligé de fournir une mise en mémoire tampon spécifique pour plusieurs requêtes **reqDcrMsgSend** simultanées en attente. Une implémentation de **Micro client** sécurisée doit utiliser le message **resDcrMsgSend** en tant qu'établissement d'une liaison de flux de contrôle.
- La fiabilité du service de transfert entre le **Micro serveur** et le **Micro client** est telle que définie pour le message **reqEncrMsgSend** au § 9.7.2.5.14.

9.7.2.6.7 Message reqDcrMsgRecv

H→C reqDcrMsgRecv(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resDcrMsgRecv(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de fournir au **Micro client** un message provenant du **Micro serveur** de la **Cible**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de déchiffrement pour laquelle le **Micro client** reçoit un message de la part du **Micro serveur**.
length: uint	Longueur du champ **msg** en octets.
msg[]: byte	Message devant être reçu de la part du **Micro serveur**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de déchiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte en mode **OnlineMsgMode**.

Préconditions de la réponse:

1) Le message a été traité par le **Micro client**, qui est prêt à accepter un nouveau message **reqDcrMsgRecv**.

Sémantique détaillée:

- Le **Micro client** doit traiter au moins un message à la fois. Le **Micro client** n'est pas obligé de fournir une mise en mémoire tampon spécifique pour plusieurs requêtes **reqDcrMsgSend** simultanées en attente; il doit toutefois prendre garde au fait qu'il est alors prêt à traiter le message suivant en raison des autres exigences de réactivité le concernant. Une implémentation d'**Hôte ECI** sécurisée doit utiliser le message **resDcrMsgRecv** en tant qu'établissement d'une liaison de flux de contrôle.

- La fiabilité du service de transfert entre le **Micro client** et le **Micro serveur** est telle que définie pour le message **reqEncrMsgSend** au § 9.7.2.5.14.

9.7.6.2.8 Message reqDcrTsData

H→C reqDcrTsData(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resDcrTsData(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de fournir au **Micro client** les données requises (très) prochainement pour le déchiffrement du contenu sur le **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de déchiffrement.
length: uint	Longueur en octets du message à transférer.
msg[]**:** byte	Message devant être transféré au **Micro client**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de déchiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte, en *mode de rechiffrement* **OfflineStream** ou **OfflineStorage**; elle utilise le *mode de format de données* **OfflineDataMode** et le *mode de synchronisation* **SyncTs**.

Préconditions de la réponse:

1) L'**Hôte ECI** est prêt à recevoir le message de données suivant.

Sémantique détaillée:

- L'**Hôte ECI** doit veiller à ce que le **Micro client** reçoive les données conformément aux exigences de synchronisation fournies par le **Micro serveur**, parallèlement au contenu chiffré devant être déchiffré.
- En mode **OfflineStorage**, le **Micro client** recevra au maximum un message de données au début de la session de déchiffrement. D'où le mode "stockage hors ligne".

9.7.2.6.9 Message reqDcrFileData

H→C reqDcrFileData(ushort **mh,** uint **datalength,** byte **data[])**
C→H resDcrFileData(ushort **mh)**

- Ce message permet à l'**Hôte ECI** de fournir au **Micro client** les données issues du **Micro serveur** de la **Cible** requises pour déchiffrer le contenu pour le **Pointeur de média**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** de la session de déchiffrement.
datalength: uint	Longueur des données en octets.
data[]**:** byte	Données destinées au **Micro client** à des fins de déchiffrement. Le format des données est opaque si le mode de format des données est **OfflineDataMode** et consiste en une boîte PSSH devant être incluse dans une boîte MOOV ou MOOF ISOBMFF si le mode de format des données est **OfflineIsobmffMode**.

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** de la session de chiffrement.

Préconditions de la requête:

1) La session de **Pointeur de média** est ouverte, en *mode de rechiffrement* **OfflineStream** ou **OfflineStorage** et en *mode de synchronisation* **SyncFile**.

Préconditions de la réponse:

1) Le **Micro client** est prêt à recevoir le message de données suivant.

Sémantique détaillée:

- L'**Hôte ECI** doit veiller à ce que le **Micro client** reçoive les données conformément aux exigences de synchronisation, parallèlement au contenu chiffré.
- L'**Hôte ECI** peut extraire une boîte PSSH d'un fichier ISOBMFF valide et la fournir au **Micro client** conformément aux exigences de synchronisation des données pour le décodage des fichiers ISOBMFF.
- En mode **OfflineStorage**, l'**Hôte ECI** fournira au maximum un message de données à la fin de la session de chiffrement. De manière générale, ce message de données doit être traité par le **Micro client** avant tout contenu.

9.7.2.6.10 Codes d'erreur de l'API de redéchiffrement par le Micro client

Les codes d'erreur de l'API de redéchiffrement par le Micro client sont répertoriés dans le Tableau 9.7.2.6.10-1.

Tableau 9.7.2.6.10-1 – Codes d'erreur liés à l'API de redéchiffrement par le Micro client

Nom	Valeur	Description
ErrDcrIpNone	-256	L'**Hôte ECI** n'a aucun(e) port/adresse IP pour communiquer avec le **Micro serveur**.

9.8 API relatives aux ressources en rapport avec les propriétés de contenu

9.8.1 Liste des API définies dans le § 9.8

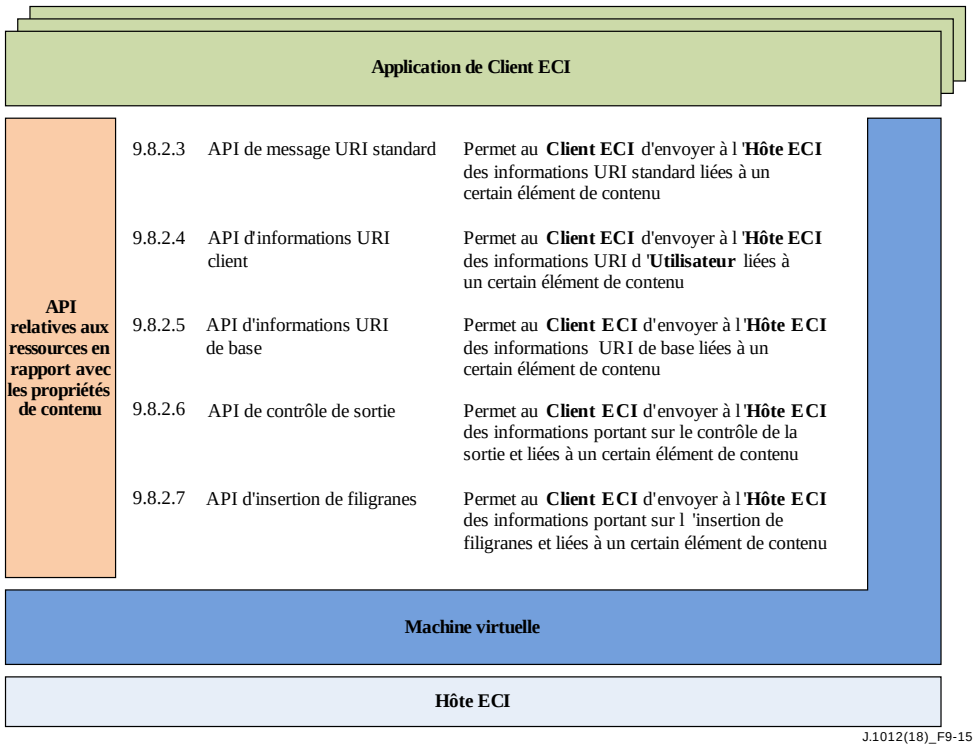


Figure 9.8.1-1 – Représentation des API définies au § 9.8

Le Tableau 9.8.1-1 répertorie les API présentées dans le § 9.8 et la Figure 9.8.1-1 illustre leur positionnement dans l'**architecture ECI**.

Tableau 9.8.1-1 – API relatives aux ressources en rapport avec la protection du contenu

Paragraphe	Nom de l'API	Description
9.8.2.3	API de message URI standard	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** (et inversement) des informations URI standard liées à un certain élément de contenu.
9.8.2.4	API d'informations URI client	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** (et inversement) des informations URI d'**Utilisateur** liées à un certain élément de contenu.
9.8.2.5	API d'informations URI de base	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** (et inversement) des informations URI de base liées à un certain élément de contenu.
9.8.2.6	API de contrôle de sortie	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** (et inversement) des informations portant sur le contrôle de la sortie et liées à un certain élément de contenu.
9.8.2.7	API d'insertion de filigranes	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** (et inversement) des informations portant sur l'insertion de filigranes et liées à un certain élément de contenu.
9.8.2.8	API de contrôle parental	Permet au **Client ECI** d'envoyer à l'**Hôte ECI** des informations portant sur les obligations relatives au contrôle parental, liées à un certain élément de contenu.
9.8.2.9	API de synchronisation des propriétés de contenu	Permet la synchronisation de divers changements dans les propriétés de contenu.
9.8.2.10	API d'authentification parentale	Permet au **Client ECI** de déléguer l'authentification parentale à une fonction d'authentification parentale standard dans l'**Hôte ECI**.
9.8.2.11	API de délégation de l'authentification parentale	Permet au **Client ECI** d'annuler une requête d'authentification parentale déléguée.
9.8.2.12	API de contrôle de protection	Permet au **Client ECI** de confier à l'**Opérateur de plate-forme** un certain contrôle sur les systèmes de protection de sortie.

9.8.2 API d'accès aux ressources relatives aux droits d'utilisation et au contrôle parental

9.8.2.1 Introduction

Ce paragraphe relatif aux API d'échange entre **Client ECI** et **Hôte ECI** permet au **Client ECI** de définir de manière sécurisée les droits et les conditions s'appliquant au contenu déchiffré.

L'API relative aux droits et aux conditions précise les aspects suivants:

- URI (informations relatives aux droits d'utilisation): générées par le **Client ECI** et utilisées par l'**Hôte ECI** pour contrôler l'application du contenu sur les sorties et applications alignées sur les normes sectorielles.
- Informations URI de base: générées par le **Client ECI** et utilisées par la **sécurité évoluée** et le sous-système matériel de l'**Hôte ECI** pour définir les droits d'utilisation de base du contenu. Le **Client ECI** peut ainsi utiliser une protection matérielle solide pour les propriétés relatives aux droits de base qui doivent être appliquées au contenu.
- Contrôle de sortie: permet au **Client ECI** de bloquer de manière sélective certaines sorties qui pourraient être actives selon les conditions des informations URI mais dont l'utilisation est néanmoins considérée comme inappropriée du point de vue des droits.
- Contrôle de l'insertion de filigranes commandé par l'**Hôte ECI**: permet au **Client ECI** d'appliquer au contenu sortant des filigranes qu'il spécifie, par l'intermédiaire d'un système d'insertion de filigranes résidant sur l'**Équipement CPE**.
- Conditions de contrôle parental: permet au **Client ECI** de transférer – au système de protection vers lequel le contenu est exporté – l'obligation d'authentifier un parent pour accorder l'accès au contenu.
- Synchronisation des propriétés du contenu: permet d'identifier comme tels des changements dans les propriétés du contenu se produisant simultanément.
- La fonction d'authentification parentale peut-être effectuée directement par un **Client ECI** ou déléguée à une fonction centrale de l'**Hôte ECI** alignée sur les normes sectorielles. L'**Hôte ECI** peut alors choisir un **Client ECI** spécifique qui effectuera l'authentification parentale pour son compte. Les options de délégation visent à permettre une authentification parentale unique sur plusieurs **Clients ECI** et sur l'**Hôte ECI**.

L'application des nouvelles propriétés de droits est liée de manière sécurisée à l'application d'un nouveau mot de contrôle en vue de désembrouiller le contenu. Cela permet de garantir que les droits sont appliqués au contenu auquel ils sont associés.

Les API relatives aux propriétés du contenu possèdent un message "set" (écrire) et un message "get" (lire). Le message "set" est utilisé par les **Clients ECI** déchiffrant le contenu pour signaler les propriétés du contenu associées au prochain mot de contrôle calculé. La fonction "get" est utilisée par les **Micro serveurs** rechiffrant le contenu pour acquérir les propriétés du contenu entrant dans le but de construire les données d'authentification et de signalisation appropriées en vue de la signalisation des propriétés du contenu rechiffré.

La version de l'API signalée dans le cadre de l'API de découverte aligne concrètement la version des propriétés de contenu utilisées.

Le contexte du **Pointeur de média** de l'**Hôte ECI** tiendra à jour au moins deux valeurs pour différentes sections de contenu, et ce pour chaque propriété de contenu. Pour le déchiffrement basé sur fichier en particulier, il tiendra à jour au moins deux sections de contenu, chacune décodée avec un identificateur de clé distinct pour chaque propriété de contenu. Le Tableau 9.8.2.1-1 énumère les fonctions de l'API. Les fonctions d'API relatives aux droits sont regroupées dans des API distinctes pour permettre une gestion indépendante des versions.

Tableau 9.8.2.1-1 – Liste des messages des API relatives aux droits d'utilisation et au contrôle parental

API	Message	Type	Sens	Étiquette	Description
ApiStdUri	setDcrStdUri	set	C→H	0x0	Écrire les informations URI standard pour le contenu à désembrouiller.
ApiStdUri	getEncrStdUri	get	C→H	0x1	Lire les informations URI standard pour le contenu à rechiffrer.
ApiCustUri	setDcrCustUri	set	C→H	0x0	Écrire des informations URI personnalisées pour le contenu à désembrouiller.
ApiCustUri	getEncrCustUri	get	C→H	0x1	Lire des informations URI personnalisées pour le contenu à rechiffrer.
ApiBasicUri	setDcrBasicUri	set	C→H	0x0	Écrire les informations URI de base pour le contenu à désembrouiller.
ApiBasicUri	getEncrBasicUri	get	C→H	0x1	Lire les informations URI de base pour le contenu à rechiffrer.
ApiOC	setDcrOutputCtl	set	C→H	0x0	Écrire les restrictions relatives au contrôle de sortie pour le contenu à désembrouiller.
ApOC	getEncrOutputCtrl	get	C→H	0x1	Lire les restrictions relatives au contrôle de sortie pour le contenu à rechiffrer.
ApiDcrMark	getDcrMarkSyst	get	H→C	0x0	Lire les systèmes d'insertion de filigranes pris en charge.
ApiDcrMark	setDcrMarkMeta	set	C→H	0x1	Écrire une valeur de contrôle pour le système d'insertion de filigranes.
ApiDcrMark	getDcrMarkMeta	get	H→C	0x2	Lire une propriété du système d'insertion de filigranes.
ApiDcrMark	setDcrMarkBasic	set	C→H	0x3	Écrire la charge utile de base du filigrane pour le contenu à désembrouiller.
ApiDcrMark	setDcrMarkExt	set	C→H	0x4	Écrire la charge utile étendue du filigrane pour le contenu à désembrouiller.
ApiPar	setDcrParCtl	set	C→H	0x0	Écrire les conditions relatives au contrôle parental pour le contenu à désembrouiller.
ApiPar	getEncrParCtrl	get	C→H	0x1	Lire les conditions relatives au contrôle parental pour le contenu à désembrouiller.
ApiCpSync	setCpSync	set	C→H	0x0	Le **Client ECI** indique que l'ensemble actuel de propriétés de contenu est cohérent et peut être appliqué au contenu devant être désembrouillé par un mot de contrôle à venir.

Tableau 9.8.2.1-1 – Liste des messages des API relatives aux droits d'utilisation et au contrôle parental

API	Message	Type	Sens	Étiquette	Description
ApiCpSync	reqCpChange	req	H→C	0x1	L'**Hôte ECI** signale un changement à venir dans les propriétés du contenu devant être rechiffré.
ApiParAuth	reqParAuthChk	req	C→H	0x0	Demande à l'**Hôte ECI** d'effectuer une authentification parentale pour le compte du **Client ECI**.
ApiParAuth	reqParAuthChkCan	req	C→H	0x1	Annule une requête précédente d'authentification parentale envoyée à l'Hôte.
ApiParAuth	reqParAuthCid	req	H→C	0x2	Demande l'autorisation parentale par code PIN pour un (futur) élément de contenu à décoder. Ce message peut déclencher un dialogue d'authentification parentale.
ApiParAuthDel	reqParAuthDel	req	H→C	0x0	L'**Hôte ECI** délègue une authentification parentale à un **Client ECI**.
ApiParAuthDel	reqParAuthDelCan	req	H→C	0x1	L'**Hôte ECI** annule une requête précédente d'authentification parentale envoyée au **Client ECI**.
ApiProtCtrl	getProtSystCtrl	get	C->H	0x0	Le **Client ECI** obtient de l'**Hôte ECI** la liste des systèmes de protection de sortie ainsi que leur prise en charge des messages SRM (messages de renouvellement de système) et des services de blocage d'identificateur de dispositif.
ApiProtCtrl	reqSrmMsg	req	C->H	0x1	Le **Client ECI Client** fournit un message SRM à un système de protection de sortie.
ApProtCtrl	reqInfoDevId	req	H->C	0x2	L'**Hôte ECI** fournit l'identificateur d'un dispositif auquel un système de protection de sortie fournit un contenu protégé dans le cadre d'une session de déchiffrement.
ApiProtCtrl	reqBlockDevId	req	C->H	0x3	Le **Client ECI** fournit l'identificateur d'un dispositif auquel aucun contenu ne doit être fourni par un système de protection de sortie dans le cadre d'une session de déchiffrement.
ApProtCtrl	setBlockProtSyst	set	C->H	0x4	Le **Client ECI** indique qu'un système de protection est jugé inadapté pour protéger le contenu de la session de déchiffrement.

9.8.2.2 Aspects liés à la sécurité et synchronisation

La spécification **ECI** permet l'authentification par l'**Hôte ECI** des informations relatives aux propriétés de contenu ci-dessus afin d'éviter une manipulation non autorisée de ces informations. Ce mécanisme garantit par ailleurs que les bons paramètres relatifs aux droits sont appliqués au contenu auquel ils sont associés. Ce processus est défini dans la Recommandation [UIT-T J.1014].

En ce qui concerne les informations relatives aux propriétés du contenu, l'**Hôte ECI** peut faciliter l'authentification des informations relatives aux droits pour le compte du **Client ECI** en utilisant des clés dans le bloc de sécurité évoluée, assurant ainsi le niveau d'intégrité le plus élevé pour l'authentification. Les **Clients ECI** doivent utiliser les services de sécurité évoluée de l'**Hôte ECI** de manière appropriée à cet effet. Ce processus est également défini dans la Recommandation [UIT-T J.1014].

Si les propriétés du contenu nécessitent que des propriétés de protection de sortie spécifiques soient appliquées à une sortie, mais que ces propriétés de protection de sortie (ou des versions plus sûres ou limitées de celles-ci) ne peuvent pas être fournies par l'**Hôte ECI**, celui-ci ne doit pas sortir le contenu et envoyer un message approprié à l'**Utilisateur**. De plus amples détails doivent être fournis dans le cadre d'un régime de conformité de l'**Écosystème ECI**.

9.8.2.3 API de message URI standard

9.8.2.3.1 Message setDcrStdUri

C→H setDcrStdUri(ushort **mh,** byte **keyId**[MaxUuidLen], StdUri **stdUri**)

- Ce message définit les informations URI standard associées à l'identificateur **keyId** sur **uri**.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
stdUri: StdUri	Informations URI standard pour le contenu comme définies dans le Tableau 9.8.2.3.1-1. La sémantique des champs correspond à celle définie dans les documents [ETSI TS 103 205] et [b-CI Plus].

Tableau 9.8.2.3.1-1 – Spécification de type pour les informations URI standard

```
typedef struct StdUri {

   uint MajorVersion: 4;
   uint tmc: 1;   /* trick_mode_control_info dans [CI+ v1.4] */
   uint reserved1: 3;
   uint aps: 2; /* aps_copy_control_info dans [CI+ v1.4] */
   uint emi: 2; /* emi_copy_control_info dans [CI+ v1.4] */
   uint ict: 1; /* ict_copy_control_info dans [CI+ v1.4] */
   uint rct: 1; /* rct_copy_control_info dans [CI+ v1.4] */
   uint reserved2: 1; /* bit réservé */
   uint dot: 1; /* dot_copy_control_info dans [CI+ v1.4] */
   uint rl: 8;     /* rl_copy_control_info dans [CI+ v1.4] */

} StdUri;

Les règles suivantes s'appliquent (les expressions excédant le champ
s'évalueront comme True) conformément au document [CI+ v1.4]:
emi == 0b00 || rct == 0b0
emi == 0b11 || (dot == 0b0 && rl == 0x00)
emi == 0b01 || tmc == 0b0

Pour la définition ci-dessus, le champ protocol_version est défini sur la
valeur 0x03; les autres valeurs sont réservées à une utilisation future.
```

Sémantique du champ StdUri:

MajorVersion: uint: 4	Version majeure de ces informations URI standard. Les **Clients ECI** mettront MajorVersion à 0b0000. Les **Hôtes ECI** implémenteront toutes les versions jusqu'à leur niveau de conformité pour ce champ, et interpréteront toute valeur élevée comme des informations URI non implémentées, auquel cas aucun droit d'utilisation ne s'appliquera.
reserved1: uint: 3	Bits réservés. Seront mis à 0b000 par le **Client ECI** et seront ignorés par les **Hôtes ECI** conformes à cette version de stdUri.
reserved2: uint: 1	Bit réservé. Sera mis à 0b0 par le **Client ECI** et sera ignoré par les **Hôtes ECI** conformes à cette version de stdUri.
Autres champs	La sémantique est telle que définie pour les champs indiqués des informations URI CI Plus v1.4 [ETSI TS 103 205] dans la définition de structure ci-dessus.

Sémantique détaillée:

- Pour le mode de désembrouillage du flux de transport, les informations URI s'appliqueront au contenu devant être décodé au moyen des clés s'appliquant à la prochaine clé de

déchiffrement. Le § 8.2.4.7 de la Recommandation [UIT-T J.1014] définit en détail le calcul de la clé de déchiffrement.

- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.3.2 Message getEncrStdUri

C→H StdUri getEncrStdUri(ushort **mh,** byte **keyId**[MaxUuidLen])

- Ce message définit les informations URI standard pour le contenu à venir.

Définition de la propriété:

- Les informations URI standard sont telles que définies dans le Tableau 9.8.2.3.1-1.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId: byte[MaxUuidLen]	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.

9.8.2.4 API d'informations URI client

9.8.2.4.1 Message setDcrCustUri

C→H setDcrCustUri(ushort **mh,** byte **keyId**[MaxUuidLen], uint **custUriLen**, byte ***custUri)**

- Ce message définit les informations URI personnalisées associées à l'identificateur **keyId** sur **uri**.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId: byte[MaxUuidLen]	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
custUriLen: uint	Longueur en octets du champ d'informations URI personnalisées.
custUri: byte *	Informations URI personnalisées pour le contenu comme définies dans le Tableau 9.8.2.4.1-1. Les octets 0 et 1 constitueront les octets msB et lsB du format d'informations URI personnalisées. Toutes les valeurs des octets 0 et 1 sont réservées, à l'exception de 0x80 et 0x00, respectivement, correspondant à une signification spécifique à l'application pour les octets suivants.

Tableau 9.8.2.4.1-1 – Spécification de type pour les informations URI personnalisées

Nom	Valeur des octets 0 et 1	Description
CustUriPrivate	0x80, 0x00	La signification des octets suivant l'octet 1 est privée. L'interprétation du reste du champ est définie par d'autres communications entre le **Client ECI** et le **Micro serveur** ou le système de protection.
RFU	Autre	Réservé à une utilisation future.

Sémantique détaillée:

- Pour le mode de désembrouillage du flux de transport, les informations URI s'appliqueront au contenu devant être décodé au moyen des clés s'appliquant à la prochaine clé de déchiffrement. Le paragraphe 8.2.4.7 de la Recommandation [UIT-T J.1014] définit en détail le calcul de la clé de déchiffrement.

- Quatre URI personnalisées distinctes peuvent être définies au maximum pour un mot de contrôle.
- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.4.2 Message getEncrCustUri

C→H custUri getEncrCustUri(ushort **mh,** byte **keyId**[MaxUuidLen], unit **custUriMaxLen)**

- Ce message obtient les informations URI personnalisées pour le contenu à venir.

Définition de la propriété:

- Les informations URI personnalisées sont telles que définies dans le Tableau 9.9.1-1.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId: byte[MaxUuidLen]	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
custUriMaxLen: uint	Longueur maximale (en octets) des informations URI personnalisées obtenues; tout contenu supplémentaire sera tronqué.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.

9.8.2.5 API d'informations URI de base

9.8.2.5.1 Message setDcrBasicUri

C→H setDcrBasicUri(ushort **mh,** byte **keyId**[MaxUuidLen], BasicUri **basicUri**)

- Ce message définit les informations URI de base associées à l'identificateur **keyId** sur **basicUri**. Les informations URI de base permettent une gestion des droits simplifiée mais hautement robuste pour le contenu déchiffré.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
basicUri: BasicUri	Informations URI de base pour le contenu comme définies dans le Tableau 9.8.2.5.1–1. La sémantique des champs correspond à celle définie dans le document [ETSI TS 103 205].

Tableau 9.8.2.5.1-1 – Spécification de type pour les informations URI de base

```
typdef byte  BasicUri;
```

Nom	Bits	Description
BasicUriVersion	7	Version majeure des informations URI de base. Si l'**Hôte ECI** n'a pas implémenté la version, il n'autorisera pas le déchiffrement ni l'utilisation du contenu. La valeur 0b0 correspond à la version 0. Toutes les autres valeurs sont réservées et ne sont pas autorisées.
BasicUriV0_0Ext	2,6	Réservé à une utilisation future, non utilisé dans la version v0.0. La seule valeur définie pour ce champ est 0b00000. Les autres valeurs ne sont pas autorisées. Les **Hôtes ECI** implémentant uniquement la version v0.0 des informations URI de base ignoreront les valeurs de ce champ: il pourra être utilisé pour les futures extensions de v0.0 compatibles avec les versions antérieures, par exemple pour assouplir le contrôle des droits par rapport à la version v0.0.
BasicUriV0_0	0,1	Version 0.0 des informations URI de base. Les valeurs et les significations de ce champ sont définies dans le Tableau 9.8.2.5.1-2.

Tableau 9.8.2.5.1-2 – Définition de la version v0.0 des informations URI de base

Nom	Valeur	Description
NoBasicProtection	0b00	Aucun contrôle des droits par l'intermédiaire des informations URI de base
RedistributionProtected	0b01	Chiffrement activé, prévention contre les réexécutions désactivée
ViewOnly	0b10	Chiffrement et prévention contre les réexécutions activés
ViewOnlyStrict	0b11	Chiffrement et prévention contre les réexécutions activés, sortie limitée à des sorties spécifiquement qualifiées (sécurisées)

Sémantique détaillée:

- Pour le mode de désembrouillage du flux de transport, les informations URI s'appliqueront au contenu devant être décodé au moyen des clés s'appliquant à la prochaine clé de déchiffrement. Le paragraphe 8.2.4.7 de la Recommandation [UIT-T J.1014] définit en détail le calcul de la clé de déchiffrement.
- Les informations URI de base permettent au **Client ECI** de contrôler l'implémentation de droits au niveau le plus élevé de protection pris en charge par l'**Hôte ECI**. Ce contrôle est possible par l'intermédiaire de deux systèmes de protection: le chiffrement, qui garantit que le contenu est en permanence embrouillé sur quelque sortie ou quelque support de stockage que ce soit; et la prévention contre les réexécutions, qui garantit que le contenu chiffré ne peut être désembrouillé que sur une connexion active (c'est-à-dire qu'il ne peut pas être enregistré). Pour plus de détails, voir la Recommandation [UIT J.1015].
- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.5.2 Message getEncrBasicUri

C→H BasicUri getEncrBasicUri(ushort **mh,** byte **keyId**[MaxUuidLen])

- Ce message obtient les informations URI de base pour le contenu à venir.

Définition de la propriété:

- Les informations URI de base sont telles que définies dans le Tableau 9.8.2.5.1-1.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.

9.8.2.6 API de contrôle de sortie

9.8.2.6.1 Message setDcrOutputCtl

C→H setDcrOutputCtl(ushort **mh,** byte **keyId**[MaxUuidLen], ushort **ocVector**)

- Ce message définit les paramètres de contrôle de sortie associés à l'identificateur **keyId** sur **ocVector**.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
ocVector: ushort	Vecteur du contrôle de sortie pour les sorties standard, tel que défini dans le Tableau 9.8.2.6.1-1.

Tableau 9.8.2.6.1-1 – Spécification du vecteur de contrôle de sortie

Nom	Bits	Description
MajorVersion	15	Version du paramètre ocVector. La valeur 0b0 est définie pour la version 1. Toutes les autres valeurs sont réservées et ne sont pas autorisées. Si un **Hôte ECI** mettant en œuvre la **version majeure** 1 reçoit une valeur autre que 0xb0, aucune sortie n'est autorisée.
OcAnyOther	14	Toute autre sortie de l'**Hôte ECI** ne correspondant à aucun des critères de qualification de sortie répertoriés ci-dessous. Si la valeur est égale à 0b0, la sortie est autorisée sur ces sorties; si la valeur est 0b1, elle est interdite. ***La valeur de ce bit modifie le codage des champs ci-après.*** Si la valeur est égale à 0b0, les restrictions ci-dessous liées à la sortie s'appliqueront. Si la valeur est égale à 0b1, le codage se fera selon une inversion binaire. P. ex., si OcAnyOther==0b1 et OcIP==0b1, la sortie sur connexion IP est autorisée. Voir la NOTE 2.
OcIP	0	La sortie sur une connexion IP est autorisée si la valeur est égale à 0b0, et interdite si la valeur est égale à 0b1.
OcUSB	1	La sortie sur une connexion USB est autorisée si la valeur est égale à 0b0, et interdite si la valeur est égale à 0b1. La condition à cela est que le contenu déchiffré ne soit pas protégé par un système de protection de sortie reconnu par l'interface **ECI** et/ou par un **Système Micro DRM ECI** sous le contrôle du **Client ECI** effectuant le déchiffrement.
OcDtcpIp	2,3	La sortie sur une connexion DTCP-IP protégée est autorisée si la valeur est égale à 0b0, et interdite si la valeur est égale à 0b1.
OcHdcp	3,4	Toute sortie protégée par HDCP. Lorsque OcAnyOther est égal à 0b0: • Valeur 0b00: sortie protégée par HDCP autorisée. • Valeur 0b01: si la version HDCP est inférieure à 2.2, la sortie sera interdite; si elle est supérieure ou égale à 2.2, la sortie sera autorisée. • Valeur 0b10: valeur réservée, non autorisée. Les **Hôtes ECI** interpréteront cette valeur comme étant égale à 0b11. • Valeur 0b11: aucune sortie protégée par HDCP autorisée. Lorsque OcAnyOther est égal à 0b1: • Valeur 0b00: aucune sortie HDCP autorisée. • Valeur 0b01: réservée; les **Hôtes ECI** interpréteront cette valeur comme étant égale à 0b00. • Valeur 0b10: si la version HDCP est supérieure ou égale à 2.2, la sortie sera autorisée, si elle est inférieure à 2.2, la sortie sera interdite. • Valeur 0b11: toute sortie protégée par HDCP est autorisée. Une version de HDCP supérieure ou égale à 2.2 signifie qu'aucune version de HDCP inférieure à 2.2 ne doit être appliquée au contenu, c'est-à-dire qu'aucune sortie vers un répéteur compatible HDCP1.x, HDCP2.0 ou HDCP2.1 ou un dispositif compatible HDCP1.x n'est autorisée. Voir "flux de contenu de type 1" tel que défini dans [b-HDCP2.3].

Tableau 9.8.2.6.1-1 – Spécification du vecteur de contrôle de sortie

Nom	Bits	Description
OcWm	5	Si la valeur de ce bit est égale à 0b1, la sortie de l'élément de contenu décodé est autorisée uniquement si l'**Équipement CPE** insère un filigrane dans l'élément. Voir la NOTE 3.
OcDtcp	6,7	Toute sortie protégée par DTCP. Lorsque OcAnyOther est égal à 0b0: • Valeur 0b00: sortie protégée par DTCP autorisée. • Valeur 0b01: si la version DTCP est inférieure à 2, la sortie sera interdite; si elle est supérieure ou égale à 2, la sortie sera autorisée. • Valeur 0b10: valeur réservée, non autorisée. Les **Hôtes ECI** interpréteront cette valeur comme étant égale à 0b11. • Valeur 0b11: aucune sortie protégée par DTCP autorisée. Lorsque OcAnyOther est égal à 0b1: • Valeur 0b00: aucune sortie DTCP autorisée. • Valeur 0b01: réservée; les **Hôtes ECI** interpréteront cette valeur comme étant égale à 0b00. • Valeur 0b10: si la version DTCP est supérieure ou égale à 2, la sortie sera autorisée, si elle est inférieure à 2, la sortie sera interdite. • Valeur 0b11: toute sortie protégée par DTCP est autorisée.
OCDownResHDCP1	8	La sortie de contenu sur une sortie protégée par HDCP1.x est autorisée si la valeur du champ **OcHdcp** est égale à 0b01 et si le contenu est ramené à 720p ou moins dans le cas où la valeur de ce champ est égale 0b0; la sortie est interdite si la valeur de ce champ est égale à 0b1.
réservé	9-13	La valeur de ce champ doit être mise à 0b00000 par les **Clients ECI** conformes à cette version de la spécification. Une implémentation d'**Hôte ECI** conforme à cette version de la spécification peut ignorer ce champ.
NOTE 1 – Le contrôle de sortie analogique est couvert par les champs **dot** et **ict** des informations URI standard. NOTE 2 – Concrètement, OcAnyOther bascule le champ de contrôle de sortie entre une liste noire de sorties (lorsque la valeur est égale à 0b0) et une liste blanche de sorties (lorsque la valeur est égale à 0b1). Si un champ de sortie a la valeur 0b1, il est "sur la liste". NOTE 3 – Les systèmes d'insertion de filigranes adaptés à cette application peuvent être soumis à approbation. Les **Hôtes ECI** dotés d'une capacité de radiodiffusion ou de multidiffusion doivent prendre en charge l'insertion de filigranes. Dans le cadre de la définition de l'application d'un système d'insertion de filigranes à un équipement CPE basé sur l'ECI, il doit être possible d'identifier de manière univoque la puce électronique, par exemple en récupérant l'identificateur de la puce électronique à partir du filigrane.		

Si plusieurs champs ocVector s'appliquent à une sortie (par exemple, une sortie IP protégée par DTCP-IP), la condition la plus restrictive est appliquée.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.6.2 Message getEncrOutputCtrl

C→H uint getEncrOutputCtrl(ushort **mh,** byte **keyId**[MaxUuidLen])

- Ce message obtient le contrôle de sortie pour le contenu à venir.

Définition de la propriété:

- Le contrôle de sortie est tel que défini dans le Tableau 9.8.2.6.1-1.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.

9.8.2.7 API d'insertion de filigranes

9.8.2.7.1 Généralités

L'API d'insertion de filigranes permet aux **Clients ECI** de découvrir les systèmes d'insertion de filigranes intégrés disponibles par l'intermédiaire de l'**Hôte ECI**, puis de lancer un dialogue de contrôle de la "configuration" avec ces systèmes. Les systèmes d'insertion de filigranes peuvent n'être en mesure d'établir un dialogue qu'avec un nombre limité de **Clients ECI**, et d'insérer un filigrane que sur un nombre limité de sessions de **Pointeur de média** simultanément.

Les systèmes d'insertion de filigranes peuvent souhaiter établir un dialogue avec des **Clients ECI** autorisés. Ces autorisations peuvent être notamment mises en place au moyen des messages setMarkMeta et getMarkMeta et d'un dialogue d'autorisation défini par le système d'insertion de filigranes.

Les **Clients ECI** peuvent réserver l'accès à un système d'insertion de filigranes en établissant un dialogue avec succès. Le **Client ECI** (identifié par son identificateur) restera en dialogue avec le système d'insertion de filigranes jusqu'à ce que celui-ci soit retiré de l'**Équipement CPE** ou qu'il se désengage.

9.8.2.7.2 Message getDcrMarkSyst

C→H MarkSystDescr **getDcrMarkSyst()**

- Ce message permet au **Client ECI** de lire les descripteurs des systèmes d'insertion de filigranes disponibles.

Définition de la propriété:

Le type de résultat MarkSystDescr sera conforme à la définition énoncée dans le Tableau 9.8.2.7.2-1.

Tableau 9.8.2.7.2-1 – Définition du type MarkSystDescr

```
#define MaxMarkSystDescr 16;

typedef ushort MarkId;  /* Identificateur d'insertion de filigranes ECI
attribué à un système d'insertion de filigranes */
// valeurs de markId: 0x8xxx est utilisé pour les systèmes d'insertion de
filigranes propriétaires.
//                0x0000 signifie qu'il n'y a aucun système d'insertion de
filigranes.
//                Toutes les autres valeurs sont réservées par l'interface
ECI, l'attribution
//                 de nouveaux identificateurs et leur publication sont
définies ailleurs.

typedef struct MarkSystDescrElem {
   MarkID markId;        /* Identificateur du système d'insertion de
filigranes */
```

```
   uchar  nrClients;     /* Nombre de clients pouvant encore être pris en
charge */
   uchar markSystFlags  /* Champ tel que défini ci-dessous */
} MarkSystDescr[MaxMarkSystDescr];
// Tous les systèmes d'insertion de filigranes disponibles seront énumérés
// comme premiers éléments de MarkSystDescr. Les éléments restants
// utiliseront markId==0x0000.


// markSystFlags:
//   Le bit 0 indique si une autorisation est requise (0b1) ou non (0b0).
//   Le bit 1 indique si le flux embrouillé est pris en charge (0b1) ou non
// (0b0).
//   Le bit 2 indique si plusieurs flux simultanés sont pris en charge (0b1)
// ou non (0b0).
//   Tous les autres bits sont réservés et seront ignorés par les clients
//   conformes à la présente Recommandation.
```

9.8.2.7.3 Message setDcrMarkMeta

C→H setDcrMarkMeta(MarkID **markId,** uchar **index,** byte **data[32])**

- Ce message permet à l'**Hôte ECI** de définir des (méta)données de contrôle pour un système d'insertion de filigranes.

Définition des paramètres:

markId: MarkId	Identificateur du système d'insertion de filigranes pour établir la définition de propriété.
index: uchar	Sous-propriété devant être établie pour les systèmes d'insertion de filigranes.
data[32]: byte	Valeur à appliquer à la sous-propriété indiquée par index.

9.8.2.7.4 Message getDcrMarkMeta

C→H byte[32] getDcrMarkMeta(MarkID **markId**, uchar **index**)

- Ce message permet au **Client ECI** d'obtenir des (méta)données de contrôle pour un système d'insertion de filigranes.

Définition de la propriété:

- Métadonnées pour le système d'insertion de filigranes doté de l'identificateur **markId** ayant la sous-propriété **index**.

Définition des paramètres:

markId: MarkId	Identificateur du système d'insertion de filigranes pour lire la définition de propriété: le type de résultat MarkSystDescr doit être conforme à la définition du Tableau 9.8.2.7.4–1.
index: uchar	Sous-propriété du système d'insertion de filigranes à lire.

9.8.2.7.5 Message setDcrMarkBasic

C→H setDcrMarkBasic(ushort **mH,** byte **keyId**[MaxUuidLen], MarkID **markId**, byte **data**[16])

- Ce message permet au **Client ECI** de définir 128 bits maximum de données utilisées pour insérer un filigrane dans le contenu devant être désembrouillé au moyen de la clé désignée.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
markId: MarkId	Identificateur du système d'insertion de filigranes.
data[16]**:** byte	Valeur de 128 bits.

9.8.2.7.6 Message setDcrMarkExt

C→H setDcrMarkExt(ushort **mH,** byte **keyId**[MaxUuidLen], ushort **markId**, uint **dataLen**, byte **data**[]**)**

- Ce message permet au **Client ECI** de définir une charge utile étendue pour le système d'insertion de filigranes en vue de l'insertion d'un filigrane dans le contenu devant être désembrouillé au moyen de la clé désignée.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId: byte[MaxUuidLen]	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
markId: ushort	Identificateur du système d'insertion de filigranes à utiliser pour insérer un filigrane dans le contenu.
datalen: uint	Longueur du champ de données.
Data[]**:** byte	Données de la charge utile pour le système d'insertion de filigranes.

9.8.2.8 API de contrôle parental

9.8.2.8.1 Message setDcrParCtl

C→H setDcrParCtl(ushort **mH,** byte **keyId**[MaxUuidLen], ParCond **pC)**

- Ce message permet au **Client ECI** de définir les conditions de contrôle parental (**pC**) pour le contenu du **Pointeur de média** mH devant être désembrouillé au moyen de la clé désignée.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les conditions de contrôle parental **pC** en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.
pC: ParCond	Conditions de contrôle parental à appliquer au contenu. Voir le Tableau 9.8.2.8.1-1 pour la définition de ParCond.

Tableau 9.8.2.8.1-1 – Spécification de type pour le contrôle parental

```
typdef struct ParCond {
   byte basicCondition; /* voir le Tableau 9.8.2.8.1-2 */
   byte extendedQualifier[16];
} ParCond;
```

Tableau 9.8.2.8.1-2 – Définition des conditions de base relatives au contrôle parental

Nom	Bits	Description
AuthRequired	7	0b1 signifie que l'authentification parentale est requise avant de restituer le contenu. 0b0 signifie que l'authentification parentale peut être requise en fonction du champ extendedQualifier.
ToggleBit	6	Ce bit alterne dans un flux pour indiquer une nouvelle exigence d'authentification parentale lorsqu'il change de valeur.
Reserved	4,5	Sera mis à 0b00.
QualifierFormat	0,3	Indique le format du champ extendedQualifier. La valeur 0x0 correspond à "aucune valeur"; le champ extendedQualifier sera alors mis à 0. La valeur 0x1 indique que le champ extendedQualifier contient un descripteur de classement parental DVB tel que défini dans le document [ETSI EN 300 468]. Les octets restants auront pour valeur 0. L'authentification sera requise même si AuthRequired==0b0, au cas où le classement requis pour le pays concerné dépasse la limite définie par le parent (comme défini par la sémantique du descripteur de classement parental DVB). Les valeurs 0x2...0xF sont réservées à une utilisation future.

Sémantique détaillée:

- L'interface **ECI** permet de transmettre les conditions d'authentification parentale avec le contenu, en tant qu'obligation, à un système protégeant le contenu désembrouillé.
- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.8.2 Message getEncrParCtrl

C→H ParCond getEncrParCtrl(ushort **mh,** byte **keyId**[MaxUuidLen])

- Ce message permet au **Client ECI** d'obtenir les conditions de contrôle parental applicables au contenu à venir.

Définition de la propriété:

- Les informations URI relatives au contrôle parental sont définies dans le Tableau 9.8.2.8.1-2.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.

9.8.2.9 API de synchronisation des propriétés de contenu

9.8.2.9.1 Message setCpSync

C→H setCpSync(ushort **mH,** byte **keyId**[MaxUuidLen]**)**

- Ce message signale à l'**Hôte ECI** que les propriétés de contenu de la section de contenu à venir indiquée par keyId seront définies par l'intermédiaire des API d'informations URI standard, d'informations URI personnalisées, d'informations URI de base, de contrôle de sortie, d'insertion de filigranes et de contrôle parental.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à décoder.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les conditions de contrôle parental **pC** en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Ce message donne à l'**Hôte ECI** le signal pour qu'il se prépare aux changements à venir dans les propriétés de contenu, y compris en envoyant un message reqCpChange à tout **Micro serveur** ayant une **Connexion d'importation/exportation** avec la session de **Pointeur de média** en cours.
- Le **Client ECI** doit être en mode de déchiffrement.

9.8.2.9.2 Message reqCpChange

H→C reqCpChange(ushort **mh,** byte **keyId**[MaxUuidLen])

- Ce message donne le signal au **Micro serveur** de se préparer à un changement de propriété de contenu en fonction des valeurs futures les plus récentes des propriétés du contenu déchiffré qui sera rechiffré par le **Micro serveur**.

Définition de la propriété:

- Les informations URI relatives au contrôle parental sont définies dans le Tableau 9.8.2.8.1-2.

Définition des paramètres:

mH: ushort	**Pointeur de média** du contenu à chiffrer.
keyId[MaxUuidLen]**:** byte	Identificateur de clé sous forme d'UUID dans l'ordre des octets du réseau, auquel s'appliquent les informations URI en cas de décodage basé sur fichier, l'octet 0 portant la valeur 0x00 (pair) ou 0x01 (impair) pour les flux au format TS afin d'indiquer l'applicabilité au mot de contrôle suivant.

Sémantique détaillée:

- Le **Client ECI** doit être en mode de chiffrement.
- Le **Client ECI** recevra les propriétés du contenu à venir lié à KeyId dans le flux déchiffré et préparera une nouvelle configuration de chiffrement pour le nouveau contenu (peut nécessiter un nouveau mot de contrôle).

9.8.2.10 API d'authentification parentale

9.8.2.10.1 Généralités

L'authentification en vue de l'approbation parentale peut être réalisée directement par un **Client ECI** au moyen d'une session d'interface homme-machine. Un **Client ECI** peut aussi demander à l'**Hôte ECI** de réaliser (ou d'avoir réalisé) l'authentification parentale, afin d'harmoniser la gestion des codes PIN et d'améliorer l'expérience **Utilisateur** en intégrant les demandes de code PIN de manière naturelle dans l'interface d'**Utilisateur** de l'**Hôte ECI**. L'**Utilisateur** peut à son tour, par l'intermédiaire de l'**Hôte ECI**, sélectionner un **Client ECI** parmi plusieurs candidats disponibles afin qu'il réalise l'authentification parentale au moyen de l'API de délégation d'authentification parentale définie au § 9.8.2.11. Cette fonction est utile lorsqu'un **Client ECI** traitant de nombreux éléments de contenu ne peut pas déléguer son authentification parentale mais peut effectuer une authentification parentale pour le compte de l'**Hôte ECI**.

Cette API permet également au **Client ECI** de lancer une authentification parentale pour un élément de contenu avant l'ouverture d'une session de média, par exemple en vue de l'authentification parentale d'un événement d'enregistrement futur.

9.8.2.10.2 Fonction d'authentification parentale standard

Ce paragraphe définit un ensemble d'exigences pour une fonction de classement parental standard basée sur des codes PIN de quatre caractères, qu'un **Hôte ECI** sera capable d'exécuter si un **Client ECI** lui en fait la demande, ou qu'un autre **Client ECI** exécutera pour le compte de l'**Hôte ECI** s'il propose ce type de service par l'intermédiaire de l'API de délégation d'authentification parentale.

Un **Hôte ECI** ou un **Client ECI** pourra fournir une autre fonction d'authentification que celle décrite dans la suite du présent paragraphe si elle garantit au minimum le même niveau d'intégrité de l'authentification parentale que le mécanisme décrit dans le présent paragraphe.

Les fonctionnalités suivantes s'appliquent au mécanisme d'authentification parentale standard basée sur code PIN:

1) L'authentification parentale est basée sur un code PIN d'au moins quatre caractères alphanumériques issus d'un ensemble d'au moins dix caractères (par exemple, chiffres).
2) Les paramètres relatifs au code PIN seront protégés par le code PIN lui-même ou par un mécanisme d'authentification maîtresse protégeant l'accès à des ressources ou services de valeur matérielle jugés particulièrement inappropriés pour des mineurs et dont le contenu doit être inaccessible à ces derniers.
3) Tous les paramètres de limite de classement parental applicables seront protégés par le code PIN ou par un mécanisme d'authentification maîtresse conformément au point 2) ci-dessus.
4) Les exigences relatives à un éventuel mécanisme d'authentification maîtresse créeront une intégrité d'authentification équivalente au moins à celle du mécanisme de code PIN défini au présent paragraphe sans qu'il soit basé sur un mécanisme d'authentification maîtresse.
5) Lors de l'achat d'un hôte, le code PIN initial destiné au classement parental ou la méthode d'authentification au moyen de l'authentification maîtresse sera transmis(e) au propriétaire uniquement.
6) Lors de l'installation d'un nouveau client, l'**Opérateur** transmettra au propriétaire uniquement le code PIN initial ou la méthode d'authentification au moyen de l'authentification maîtresse.
7) Le **Fabricant** ou un dépositaire agissant pour son compte pourra fournir une méthode de réinitialisation du code PIN à sa valeur initiale ou un service permettant au propriétaire de demander une nouvelle valeur pour le code PIN, dont il sera le seul destinataire.
8) L'**Opérateur** pourra fournir une méthode de réinitialisation du code PIN à sa valeur initiale ou un service permettant au propriétaire de demander une nouvelle valeur pour le code PIN, dont il sera le seul destinataire.
9) Si cinq tentatives d'authentification successives échouent en l'espace de 15 minutes, la fonction d'authentification parentale refusera d'effectuer une nouvelle authentification pendant au moins 15 minutes.
10) Il ne sera pas possible de récupérer ou de réinitialiser le code PIN au moyen d'un logiciel **Utilisateur** courant, d'applications téléchargées s'exécutant sur l'**Équipement CPE**, ou de toute interface d'**Utilisateur** ou courante.

9.8.2.10.3 Message reqParAuthChk

C→H reqParAuthChk(ushort **mH) →**
C→H resParAuthChk(ushort **mH,** bool **ok)**

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** d'effectuer une vérification d'authentification parentale au moyen de la fonction d'authentification parentale standard de l'**Hôte ECI** (voir le § 9.8.2.10) et de donner le résultat dans un message de réponse.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du contenu à décoder.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du contenu à décoder.
ok: bool	La valeur "true" signifie que l'authentification a réussi, la valeur "false" indique un autre résultat, y compris une expiration du délai.

Sémantique détaillée:

- L'**Hôte ECI** ne distinguera qu'une seule vérification d'authentification parentale en attente par **Pointeur de média**. Si une deuxième requête est émise sur le même **Pointeur de média** avant que la première ait reçu une réponse ou ait été annulée, deux **Réponses** identiques seront produites.
- **reqParAuthChk**: L'**Hôte ECI** utilisera une valeur de délai d'expiration pour la demande d'authentification parentale, qui se terminera après une période raisonnable si aucune personne n'est présente ou disposée à effectuer l'authentification, comme proposé dans le document [b-UIT-T J Suppl. 7].

9.8.2.10.4 Message reqParAuthChkCan

C→H reqParAuthChkCan(ushort **mH) →**
H→C resParAuthChkCan(ushort **mH)**

- Le **Client ECI** annule toute demande précédente d'authentification parentale envoyée à l'**Hôte ECI**.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du contenu à décoder.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du contenu à décoder.

Postconditions de la réponse:

1) L'**Hôte ECI peut** envoyer au **Client ECI** la réponse à un message **reqParAuthChk** précédent avant le message **resParAuthChkCan**, mais pas après.

9.8.2.10.5 Message reqParAuthCid

H→C reqParAuthCid(uint **cidLength,** byte **cid**[]**) →**
C→H resParAuthCid(bool **ok)**

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** d'effectuer toute authentification requise pour un élément de contenu futur identifié par **cid**.

Définition des paramètres de la requête:

cidLength: uint	Longueur du paramètre cid.
cid[]**:** byte	Identification du contenu devant faire l'objet de l'authentification parentale (le cas échéant). Le premier octet indique le format du paramètre d'identification du contenu, tel que défini dans le Tableau 9.8.2.10.5-1.

Tableau 9.8.2.10.5-1 – Formats d'identification du contenu

Nom	Valeur	Description
CidDvbEvent	0x01	Identification d'un événement DVB. Les octets suivant les octets du paramètre cid ont la valeur de la séquence suivante: identificateur du réseau d'origine (2 octets), identificateur du flux de transport (2 octets), identificateur du service (2 octets), identificateur de l'événement (2 octets), tel que défini dans la table EIT définie dans le document [ETSI EN 300 468]. Tous les champs de 2 octets de la séquence sont représentés dans l'ordre du réseau (octet de plus fort poids en premier).
RFU	autre	Réservé à une utilisation future.

Définition des paramètres de la réponse:

ok: bool	"true" si l'authentification parentale a réussi ou si elle n'est pas requise.

Sémantique détaillée:

- Le **Client ECI** conservera un enregistrement non volatile des identifications de contenu authentifiées au moyen de cette fonction. Il pourra supprimer les enregistrements les plus anciens ou qui ne seront plus nécessaires dans le futur si l'espace de stockage devient insuffisant. Des exigences minimales pour cette mise en tampon de l'identification de contenu sont proposées dans le document [b-UIT-T J Suppl. 7].

Les codes d'erreur associés sont répertoriés dans le Tableau 9.8.2.10.5-2.

Tableau 9.8.2.10.5-2 – Codes d'erreur de l'API de session de média pour les médias TS

Nom	Valeur	Description
ErrParAuthCidUnknOk	1	Il n'a pas été possible d'identifier le statut de l'authentification parentale de l'élément de contenu, mais l'authentification parentale a été réalisée et s'est avérée correcte.

Le code d'erreur ci-dessus peut également être renvoyé si l'accès aux ressources de réseau requises n'était pas disponible.

9.8.2.11 API de délégation de l'authentification parentale

9.8.2.11.1 Généralités

Cette API permet à un **Client ECI** d'indiquer qu'il peut exécuter une fonction d'authentification parentale standard telle que définie au § 9.8.2.10.2, et à l'**Hôte ECI** de déléguer les vérifications par code PIN à ce type de **Client ECI**.

Un **Client ECI** peut indiquer lors de son initialisation au moyen de l'API de configuration qu'il prend en charge l'API de délégation d'authentification.

NOTE – un **Client ECI** peut également choisir de ne pas déléguer sa propre authentification parentale, par exemple pour des raisons d'ordre commercial, juridique ou liées à la sécurité.

L'**Hôte ECI** proposera une fonction de configuration permettant à l'**Utilisateur** de sélectionner l'**Hôte ECI** devant effectuer l'authentification du contrôle parental standard ou de déléguer l'authentification du contrôle parental standard à l'un des **Clients ECI** prenant cette fonction en charge.

9.8.2.11.2 Message reqParAuthDel

H→C reqParAuthDel(ushort **mh**) →
C→H resParAuthDel(ushort **mH,** bool **ok)**

- Ce message permet à l'**Hôte ECI** de demander au **Client ECI** d'effectuer pour son compte une authentification parentale déléguée pour le contenu sur le **Pointeur de média** mH.

Définition des paramètres de la requête:

mH: ushort	**Pointeur de média** du contenu à décoder.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du contenu à décoder.
ok: bool	La valeur "true" signifie que l'authentification parentale a réussi; la valeur "false" indique qu'elle a échoué ou que le délai a expiré.

Sémantique détaillée:

- Le **Client ECI** ne distinguera qu'une seule vérification d'authentification parentale en attente par **Pointeur de média**. Si une deuxième requête est émise sur le même **Pointeur de média** avant que la première ait reçu une réponse ou ait été annulée, deux **Réponses** identiques seront produites.
- Le **Client ECI** utilisera une valeur de délai d'expiration pour la demande d'authentification parentale, qui se terminera après une période raisonnable si aucune personne n'est présente ou disposée à effectuer l'authentification, comme proposé dans le document [b-UIT-T J Suppl. 7].

9.8.2.11.3 Message setParAuthDelCan

H→C reqParAuthDelCan(ushort **mH**) →
C→H resParAuthDelCan(ushort **mH)**

- Ce message permet à l'**Hôte ECI** d'annuler une demande d'authentification parentale déléguée.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du contenu à décoder.

Définition des paramètres de la réponse:

mH: ushort	**Pointeur de média** du contenu à décoder.

Postconditions de la réponse:

- L'**Hôte ECI** peut envoyer au **Client ECI** la réponse à un message reqParAuthDel précédent avant le message resParAuthDelCan, mais pas après.

9.8.2.12 API de contrôle de système de protection

9.8.2.12.1 Introduction

Le contenu déchiffré par un **Client ECI** peut être fourni à différentes sorties de l'**Équipement CPE**. Une sortie est généralement protégée par un système de protection de sortie. Un système de protection de sortie peut permettre d'accepter les messages de renouvellement de système (SRM) émanant d'un **Client ECI** et permettre au **Client ECI** de bloquer les sorties vers les dispositifs connectés via le système de protection de sortie au cas où l'identificateur des dispositifs (dans le contexte du système de protection de sortie) serait répertorié comme compromis.

Un système de protection peut prendre en charge plusieurs sorties.

9.8.2.12.2 Message getProtSystCtrl

C->H getProtSystCtrl()

- Ce message permet au **Client ECI** de lire la liste des systèmes de protection de sortie pris en charge par l'équipement CPE, leur version ainsi que leur prise en charge des messages SRM (messages de renouvellement de système) et des services de blocage d'identificateur de dispositif.

Tableau 9.8.2.12.2-1 – Spécification de la matrice de contrôle de protection

```
typedef struct ProtCtrlElem {
   ushort protSysType;     // type de système de protection conformément au
                           // tableau sect-2
   uint   srmSupp:4;       // niveau de prise en charge des messages SRM
                           // conformément au tableau sect-3
   uint   devIdSupp:1;     // 0b0 signifie que les services relatifs à
                           // l'identificateur de dispositif ne sont pas pris
                           // en charge,
                           // 0b1 signifie que les services relatifs à
                           // l'identificateur de dispositif sont pris en charge
   uint   reserved:11;     // réservé; doit avoir la valeur 0b00000000000
} ProtCtrlElem;

#define MaxProtCtrlArr 32
typedef ProtCtrlElem ProtCtrlArr[MaxProtCtrlArr];
   // Un système de protection répertorié dans la matrice peut protéger
   // plusieurs sorties. Chaque valeur d'élément ProtCtrlElem, sauf lorque
   // protSustType=0x0000, ne doit apparaître qu'une seule fois dans la matrice
   // ProtCtrlArr. Tous les éléments ProtCtrlElem pour lesquels la valeur de
   // l'élément ProtColElem est différente de 0x0000 doivent se trouver dans les
   // éléments d'indices les plus faibles de la matrice ProtCtrlArr, ceux pour
   // lesquels la valeur est égale à à 0x0000 doivent se trouver à la fin de la
   // matrice.
```

Table 9.8.2.12.2-2 – Valeurs du type de système de protection de sortie

Nom	Valeur	Type de système de protection de sortie
OpNoProtSyst	0x0000	Pas de système de protection de sortie
OpHDCP_1	0x0010	HDCP version1
OpHDCP_21	0x0011	HDCP version 2.0 ou 2.1
OpHDCP_22	0x0012	HDCP version 2.2 ou supérieure
OpDTCP_1	0x0020	DCTP version 1
OpDTCP_2	0x0021	DTCP version 2 ou supérieure
OpDTCP_IP1	0x0030	DTCP IP
Proprietary	0x8xxx	Peut être défini en dehors du cadre de la présente spécification
Reserved	Autres valeurs	Réservé à une utilisation future

Table 9.8.2.12.2-3 – Valeurs de prise en charge des messages SRM

Nom	Valeur	Prise en charge des messages DRM
SrmNone	0x0	Pas de prise en charge des messages SRM
SrmProtSysSpecV1	0x1	Prise en charge des messages SRM conformément à la version 1 (mais pas une version supérieure) de la spécification du système de protection de sortie
SrmProtSysSpecV2	0x2	Prise en charge des messages SRM conformément à la version 2 (mais pas une version supérieure) de la spécification du système de protection de sortie
SrmProtSysSpecV3	0x3	Prise en charge des messages SRM conformément à la version 3 (mais pas une version supérieure) de la spécification du système de protection de sortie
SrmProtSysSpecV4	0x4	Prise en charge des messages SRM conformément à la version 4 (mais pas une version supérieure) de la spécification du système de protection de sortie
reserved	0x5..0xC	Peut être défini en dehors du cadre de la présente spécification
Proprietary	0xD-0xF	Réservé à une utilisation future

Sémantique:

- La prise en charge des services relatifs à l'identificateur de dispositif signifie que le système de protection doit prendre en charge l'identification et le blocage de toute connexion protégée à un dispositif au moyen des messages reqBlockDevId et resBlockDevId.
- La configuration des fonctions de protection de sortie doit être statique pendant la "durée de vie" du client.

9.8.2.12.3 Message reqSrmMsg

C→H **reqSrmMsg**(ushort **protSysType**, uint **srmLen**, byte **srmData**[]) →
H→C **resSrmMsg**()

- Ce message permet au **Client ECI** d'envoyer un message SRM au type de système de protection.

Définition des paramètres de la requête:

protSysType[]**:** ushort	Le type de système de protection auquel ce message SRM est destiné. Note : Les messages SRM peuvent s'appliquer à plusieurs types de systèmes de protection de la même famille. Dans ce cas, il suffit d'envoyer le message SRM à l'hôte une seule fois et non pour chaque type.
srmLength: uint	Longueur du message SRM
srmData:byte[]	Message SRM

Préconditions de la requête:

- Aucune message **reqSrmMsg** n'a été envoyé précédemment ou le message **resSrmMsg** de réponse au dernier message reqSrmMsg a été reçu.

Sémantique détaillée:

- L'hôte ECI doit envoyer le message **resSrmMsg** dès que possible.

Tableau 9.8.2.12.3-1 – Codes d'erreur du message reqSrmMsg

Nom	Description
ErrReqSrmMsgOverflow	Vor le § 9.8.2.12.7.

9.8.2.12.4 Message reqInfoDevId

H→C **reqInfoDevId**(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[])→
C→H **resInfoDevId**(ushort **mh**)

- Ce message permet à l'**Hôte ECI** d'indiquer les dispositifs (identifiés par **devId**) auxquels le contenu qui peut être déchiffré par le dispositif est envoyé en utilisant le système de protection **protSysType** dans le cadre de la session de déchiffrement **mh**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** pour la session de déchiffrement pour laquelle le dispositif ayant l'identificateur **devId** est utilisé.
protSysType: ushort	Système de protection utilisé pour protéger le contenu à fournir à tout dispositif ayant l'identificateur **devId** – voir le Tableau 6.4.2-1 du document [b-UIT-T J Suppl. 7]
lenDevId: uint	Longueur du champ **devId** en octets.
devId[]**:** byte	Identificateur du dispositif – le codage spécifique est défini dans une autre spécification

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** pour la session de déchiffrement pour laquelle la réponse est fournie.

Préconditions de la requête:

- Aucune message **reqInfoDevId** dans le cadre de la session **mh** n'a été envoyé précédemment ou le message **resInfoDevId** de réponse au dernier message **reqInfoDevId** dans le cadre de la session **mh** a été reçu.

Sémantique détaillée:

- L'**Hôte ECI** doit envoyer l'identificateur **devId** de chaque dispositif connecté à la sortie de la session **mh** dès que possible.

Tableau 9.8.2.12.4-1 – Codes d'erreur du message reqInfoDevId

Nom	Description
ErrReqInfoDevOverflow	Voir le § 9.8.2.12.7.

9.8.2.12.5 Message reqBlockDevId

C→H reqBlockDevId(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[])→
H→C resBlockDevId(ushort **mh**)

- Ce message permet au **Client ECI** de bloquer les dispositifs ayant l'identificateur **devId** auxquels le contenu déchiffré est envoyé en utilisant le système de protection **protSysType** dans le cadre de la session de déchiffrement **mh**.

Définition des paramètres de la requête:

mh: ushort	**Pointeur de média** pour la session de déchiffrement pour laquelle le dispositif ayant l'identificateur **devId** est utilisé.
protSysType: ushort	Système de protection utilisé pour protéger le contenu à fournir à tout dispositif ayant l'identificateur **devId** – voir le Tableau 6.4.2-1 du document [b-UIT-T J Suppl. 7]
lenDevId: uint	Longueur du champ **devId** en octets.
devId[]**:** byte	Identificateur du dispositif – le codage spécifique est défini dans une autre spécification

Définition des paramètres de la réponse:

mh: ushort	**Pointeur de média** pour la session de déchiffrement pour laquelle la réponse est fournie.

Préconditions de la requête:

- Aucun message **reqBlockDevId** dans le cadre de la session **mh** n'a été envoyé ou le message **resBlockDevId** de réponse au dernier message **reqBlockDevId** dans le cadre de la session **mh** a été reçu.

Sémantique:

- Lorsqu'il reçoit un message **reqBlockDevId** valide, l'**Hôte ECI** doit répondre par un message **ErrReqOkNoId** (voir le Tableau 9.3.4-1) et faire en sorte que la sortie vers tout dispositif ayant l'identificateur **devId** soit bloquée.

9.8.2.12.6 Messsage setBlockProtSyst

C→H setBlockProtSyst(ushort **mh**, ushort **protSysType** bool **block)**

- Ce message permet au **Client ECI** de bloquer tout contenu déchiffré envoyé en utilisant le système de protection **protSysType** dans le cadre de session de déchiffrement **mh**.

Définition des paramètres:

mh: ushort	**Pointeur de média** pour la session de déchiffrement pour laquelle le contenu doit être bloqué.
protSysType: ushort	Système de protection utilisé pour protéger le contenu à fournir à tout dispositif ayant l'identificateur **devId** – voir le Tableau 6.4.2-1 du document [b-UIT-T J Suppl. 7]
block: bool	**True** si le contenu doit être bloqué, **False** dans le cas contraire.

Sémantique:

- Si le paramètre **block** est configuré de **True** à **False** pour un système **protSysType** dans le cadre d'une session **mh**, l'**Hôte ECI** doit envoyer tous les identificateurs **devID** pour ce système **protSysType** utilisés pour la sortie dans le cadre de la session **mh** en utilisant le message **reqInfoDevId** si l'implémentation du système **protSysType** le permet (comme signalé par le message **getProtSystCtrl**).

9.8.2.12.7 Codes d'erreur pour l'API de contrôle de système de protection

- Les codes d'erreur pour l'API de contrôle de système de protection sont répertoriés dans le Tableau 9.8.2.12.7-1.

Tableau 9.8.2.12.7-1 – Codes d'erreur relatifs à l'API de contrôle de système de protection

Nom	Valeur	Description
ErrReqSrmMsgOverflow	-256	L'**Hôte ECI** indique qu'il ne peut pas encore accepter le message ReqSrmMsg suivant**.**
ErrReqInfoDevOverflow	-257	Le **Client ECI** indique qu'il ne peut pas encore accepter le message ReqInfoDev suivant**.**

9.9 API relatives à la communication entre Clients ECI et application

9.9.1 Liste des API définies dans ce paragraphe

Le Tableau 9.9.1-1 répertorie les API présentées dans ce paragraphe.

Tableau 9.9.1-1 – API relatives aux ressources en rapport avec les communications entre les clients ECI et les applications

Paragraphe	Nom de l'API	Description
9.9.2	API de communication entre clients	Permet à un **Client ECI** d'établir un trajet de communication direct avec un autre **Client ECI**.

9.9.2 API de communication entre clients

9.9.2.1 Généralités

L'**Hôte ECI** offre un environnement permettant un échange normalisé d'informations entre **Clients ECI**, sous forme d'informations d'importation/exportation, d'informations URI et de contenu. Les **Clients ECI** peuvent communiquer entre eux pour fournir des fonctionnalités supplémentaires (actuellement non définies dans le cadre de l'interface **ECI**). Les **Clients ECI** peuvent signaler par l'intermédiaire de la ressource de découverte (voir § 9.4.2) leur capacité et disposition de principe à

prendre en charge la communication entre clients. Après l'initialisation du système, ils peuvent lire les identités d'autres **Clients ECI**, y compris les **Connexions d'importation/exportation**. Les **Clients ECI** peuvent ouvrir un canal de communication (appelé "conduit") vers un homologue potentiel et échanger des messages par le biais de ce conduit. Les deux clients peuvent annuler le conduit. Le conduit d'un **Client ECI** est fermé par l'**Hôte ECI** lors de l'arrêt ou de la réinitialisation de son **Client ECI** homologue.

L'**Hôte ECI** fournit les identités des **Clients ECI** qui sont authentifiées à l'aide des **Chaînes de Certificats ECI** des **Clients ECI**. Les **Clients ECI** fourniront un mécanisme d'authentification indépendant supplémentaire au cas où la communication avec un homologue pourrait conduire à des risques de sécurité.

Dans le cas d'une communication entre un **Client ECI** décodant du contenu et un autre **Client ECI** le rechiffrant (**Micro serveur**), il est recommandé pour la configuration du conduit que celui-ci soit initié (ouvert) par le **Micro serveur**.

Le Tableau 9.9.2.1-1 répertorie les messages de l'API de communication entre clients.

Tableau 9.9.2.1-1 – Messages de l'API de communication entre clients

Message	Type	Sens	Étiquette	Description
getIccMaxClients	S	C→H	0x0	Le **Client ECI** lit le nombre maximal de **Clients ECI** que l'**Hôte ECI** peut prendre en charge.
reqIccSystemReady	A	H→C	0x1	L'**Hôte ECI** informe le **Client ECI** que tous les **Clients ECI** sont initialisés.
getIccClientInfo	S	C→H	0x2	Le **Client ECI** lit l'identité et le statut de connexion d'un autre **Client ECI** dans le système.
reqIccPipeOpen	A	C→H	0x3	Demande d'ouverture de conduit vers un autre **Client ECI**.
reqIccPipeOpenReq	A	H→C	0x4	Demande entrante d'ouverture de conduit de la part d'un autre **Client ECI**.
reqIccPipeCancel	A	C→H	0x5	Le **Client ECI** annule le conduit.
reqIccPipeClose	A	H→C	0x6	L'**Hôte ECI** informe le **Client ECI** que le conduit avec son homologue a été fermé.
reqIccPipeMsgSend	A	C→H	0x7	Le **Client ECI** envoie un message à son homologue de conduit.
reqIccPipeMsgRecv	A	H→C	0x8	Le **Client ECI** reçoit un message de la part de son homologue de conduit.

9.9.2.2 Message getIccMaxClients

C→H uint getIccMaxClients()

- Obtient le nombre maximal de **Clients ECI** que l'**Hôte ECI** peut prendre en charge.

Définition de la propriété:

- Entier non signé représentant le nombre maximal de **Clients ECI** que l'**Hôte ECI** peut prendre en charge.

9.9.2.3 Message reqIccSystemReady

H→C reqIccSystemReady()

- L'**Hôte ECI** informe le **Client ECI** que tous les autres **Clients ECI** sont initialisés.

Sémantique:

- Ce message est fourni à l'initialisation du système pour indiquer à tous les **Clients ECI** enregistrés auprès de cette API qu'ils peuvent commencer à lire le registre d'information des clients et ouvrir des conduits vers d'autres **Clients ECI**.
- Le champ ConnId du résultat reflète le dernier statut des **Connexions d'importation/exportation** du **Client ECI** avec un homologue potentiel. Elles peuvent être amenées à changer.

- Aucun message de résultat n'est requis.

9.9.2.4 Message getIccClientInfo

C→H ClientInfo getIccClientInfo(ushort **clientId)**

- Le **Client ECI** lit l'identité et le statut de connexion d'un autre **Client ECI** dans le système.

Définition des paramètres:

clientId: ushort	Identificateur du client en vue de la configuration des conduits. Il reste identique tout au long du cycle de vie du système et change lors de la réinitialisation.

Définition de la propriété:

- L'identification de connexion est une propriété dynamique.
- ClientInfo est une structure fournissant l'identité du **Client ECI** désigné et de toutes les **Connexions d'importation/exportation** avec celui-ci, comme défini ci-dessous.

Définition du type ClientInfo:

```
#define MaxConnId 32


typedef struct ClientInfo {
  ECI_Operator_Id operatorId;
  ECI_Platform_Operation_Id platformOperationId;
  ECI_Vendor_Id vendorId;
  union {
    ECI_Client_Series_Id clientSeriesId;
    ECI_Client_Id clientId;
  } client;
  ushort connId[MaxConnId];
}
```

Définitions des champs:

operatorId: ECI_Operator_Id	Identificateur d'opérateur du **Client ECI**.
platformOperationId: ECI_Platform_Operation_Id	Identificateur d'Opération de plate-forme du **Client ECI**.
client: union	ECI_Client_Series_Id ou ECI_Client_Id. Le type de champ de clientSeriesId et de clientId détermine s'il s'agit ici d'un clientSeriesId ou d'un clientId.
VendorId: ECI_Vendor_Id	Identificateur de fournisseur du **Client ECI**.
clientSeriesId: ECI_Client_Series_Id	Identificateur de la série de clients du **Client ECI**.
clientId: ECI_Client_Id	Identificateur de client du **Client ECI**.
connId: ushort[MaxConnId]	Tableau d'identificateurs de connexion; la valeur 0xFFFF indique une entrée de tableau vide. Les entrées de tableau vides sont toutes situées à la fin du tableau.

9.9.2.5 Message reqIccPipeOpen

C→H reqIccPipeOpen(ushort **clientId,** byte **protocolId[16]) →**
H→C resIccPipeOpen(ushort **clientId)**

- Ce message permet au **Client ECI** de demander à l'**Hôte ECI** d'ouvrir un conduit vers un autre **Client ECI**.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client vers lequel un conduit est demandé.
protocolId[16]: byte	Identificateur du protocole de message à utiliser, sous forme d'UUID ([IETF RFC 4122]) dont les octets sont dans l'ordre du réseau dans le tableau.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client vers lequel l'ouverture d'un conduit a été demandée.

Préconditions de la réponse:

- Le conduit est ouvert ou un code d'erreur est renvoyé. Les codes d'erreur associés sont répertoriés dans le Tableau 9.9.2.5-1.

Tableau 9.9.2.5-1 – Codes d'erreur du message reqIccPipeOpen

Nom	Description
ErrIccPipeOpenReject	Voir le Tableau 9.9.2.11–1.
ErrIccPipeOpenNoConn	
ErrIccPipeOpenProtocol	
ErrIccPipeOpenNotReady	

9.9.2.6 Message reqIccPipeOpenReq

H→C reqIccPipeOpenReq(ushort **clientId,** byte **protocolId**[16]**) →**
C→H resIccPipeOpen(ushort **clientId)**

- Ce message permet au **Client ECI** de recevoir de la part d'un autre **Client ECI** une demande entrante d'ouverture de conduit via l'**Hôte ECI**.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client demandant l'ouverture d'un conduit.
protocolId[16]: byte	Identificateur du protocole de message à utiliser, sous forme d'UUID ([IETF RFC 4122]) dont les octets sont dans l'ordre du réseau.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client ayant demandé l'ouverture du conduit.

Sémantique:

- cliendId doit avoir la même valeur dans la réponse et dans la requête.

Préconditions de la réponse:

- Le **Client ECI** peut refuser le conduit. Les codes d'erreur sont identiques à ceux liés à l'ouverture d'un conduit et sont transmis de manière transparente au demandeur. Ils sont répertoriés dans le Tableau 9.9.2.5-1.

9.9.2.7 Message reqIccPipeCancel

C→H reqIccPipeCancel(ushort **clientId) →**
H→C resIccPipeCancel(ushort **clientId)**

- Ce message permet au **Client ECI** d'indiquer à l'**Hôte ECI** qu'il veut mettre fin au conduit.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client du conduit annulé.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client du conduit annulé.

Sémantique:

- cliendId doit avoir la même valeur dans la réponse et dans la requête.

Préconditions de la réponse:

- Le conduit est fermé: le **Client ECI** ayant demandé l'annulation du conduit ne recevra plus de messages provenant de ce dernier.

Sémantique détaillée:

- Si le conduit n'a pas été ouvert, aucune condition d'erreur n'est applicable.

9.9.2.8 Message reqIccPipeClose

H→C reqIccPipeClose(ushort **clientId,** uint **reason) →**
C→H resIccPipeClose(ushort **clientId)**

- Ce message permet à l'**Hôte ECI** d'informer le **Client ECI** que le conduit avec son homologue a été fermé.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client du conduit fermé.
reason: uint	Raison de la fermeture du conduit. Les valeurs valides sont répertoriées dans le Tableau 9.9.2.11–1.

Tableau 9.9.2.8-1 – Valeurs des raisons du message reqIccPipeClose

Nom	Valeur	Description
IccPipeCloseCancel	0x01	Le conduit a été fermé par l'homologue au moyen d'un message reqIccPipeCancel.
IccPipeCloseStop	0x02	L'**Hôte ECI** a fermé le conduit parce le **Client ECI** homologue s'est arrêté. Il se peut que le **Client ECI** soit réinitialisé par la suite.
RFU	Autre	Réservé à une utilisation future.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client du conduit fermé.

Préconditions de la requête:

- Plus aucun message ne sera envoyé via le conduit.

Préconditions de la réponse:

- Le **Client ECI** ne tentera plus d'envoyer de nouveaux messages via le conduit (fermé).

9.9.2.9 Message reqIccPipeMsgSend

C→H reqIccPipeMsgSend(ushort **clientId,** uint **msgId,** uint **dataLen,** byte **data[])→**
H→C resIccPipeMsgSend(ushort **clientId)**

- Ce message permet au **Client ECI** d'envoyer un message à son homologue de conduit. Les codes d'erreur associés sont répertoriés dans le Tableau 9.9.2.11-1.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client auquel le message est envoyé.
msgId: uint	Identificateur du message. Toutes les valeurs négatives et les zéros sont réservés; toutes les valeurs positives sont spécifiques aux applications (la signification est définie dans le contexte de l'émetteur et du destinataire).
dataLen: uint	Longueur du paramètre data en nombre d'octets. Elle ne doit pas dépasser 32 768.
data[]: byte	Champ de données du message.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client du conduit.

Préconditions de la requête:

- Le message reqIccMsgSend suivant ne peut être envoyé qu'une fois que le message resIccMsgSend précédent se rapportant au même conduit a été reçu.

Tableau 9.9.2.9-1 – Codes d'erreur du message reqIccPipeMsgSend

Nom	Description
ErrIccPipeClosed	Voir le Tableau 9.9.2.11–1.

9.9.2.10 Message reqIccPipeMsgRecv

H-→C reqIccPipeMsgRecv(ushort **clientId,** uint **msgId,** uint **dataLen,** byte **data[])→ C→H resIccPipeMsgRecv(**ushort **clientId)**

- Ce message permet au **Client ECI** de recevoir un message de la part de son homologue de conduit.

Définition des paramètres de la requête:

clientId: ushort	Identificateur du client depuis lequel le message est reçu.
msgId: uint	Identificateur du message. Toutes les valeurs négatives et les zéros sont réservés; toutes les valeurs positives sont spécifiques aux applications (la signification est définie dans le contexte de l'émetteur et du destinataire).
dataLen: uint	Longueur du paramètre data en nombre d'octets. Elle ne doit pas dépasser 32 768.
data: byte[]	Champ de données du message.

Définition des paramètres du résultat:

clientId: ushort	Identificateur du client du conduit.

Préconditions de la requête:

- Le message reqIccMsgRecv suivant ne peut être envoyé qu'une fois que le message resIccMsgRecv précédent se rapportant au même conduit a été reçu.

9.9.2.11 Codes d'erreur de l'API de communication entre clients

Les codes d'erreur de l'API de communication entre clients sont répertoriés dans le Tableau 9.9.2.11-1.

Tableau 9.9.2.11-1 – Codes d'erreur relatifs à la communication entre clients

Nom	Valeur	Description
ErrIccPipeOpenReject	-256	L'homologue a refusé le conduit.
ErrIccPipeOpenNoConn	-257	L'homologue a refusé le conduit parce qu'aucune **Connexion d'importation/exportation** n'est établie avec le **Client ECI**.
ErrIccPipeOpenProtocol	-258	L'homologue refuse le protocole proposé pour le conduit.
ErrIccPipeOpenNotReady	-259	L'homologue ne se trouve pas dans un état lui permettant d'accepter un conduit. Il est recommandé de tenter à nouveau d'établir un conduit ultérieurement.
ErrIccPipeClosed	-260	Le conduit est fermé.

10 Fonctionnalités obligatoires et facultatives de l'Hôte ECI

10.1 Introduction

Les spécifications techniques du système **ECI** permettent de prendre en charge des solutions techniques pour une très large gamme d'**Équipements CPE** destinés à la consommation de médias. Il appartient au **Fabricant** d'**Équipement CPE** de décider des fonctions frontales, centrales et dorsales qu'il souhaite implémenter sur son dispositif. En ce qui concerne les fonctionnalités frontales et dorsales d'un dispositif, il est probable que le **Fabricant** n'implémentera que les API **ECI** adaptées à son matériel/sa pile de protocoles. Afin de donner plus de flexibilité à l'**Utilisateur**, le Tableau 10.2-1 répertorie toutes les API obligatoires (O), facultatives (F) et conditionnelles (C) pour les différentes catégories d'**Équipements CPE**.

10.2 liste des fonctionnalités ECI obligatoires et facultatives pour différents types de dispositifs CPE.

Le Tableau 10.2-1 répertorie les fonctionnalités **ECI** obligatoires et facultatives pour différents types de dispositifs **CPE**. L'implémentation de plusieurs API est conditionnelle: elle dépend de la disponibilité de certains composants matériels/logiciels du dispositif **CPE**.

Tableau 10.2-1 – liste des fonctionnalités ECI obligatoires et facultatives

API	Paragraphe	Hôte	Condition (le cas échéant)	Client déchiff.	Micro serveur	Micro client
Découverte des interfaces de l'Hôte	9.4.2	O		O	O	O
Interface homme-machine	9.4.3	O		F	F	F
IP	9.4.4	C	Si la connectivité IP est prise en charge	F	F	F
HTTP(S)	9.4.4.6	O		F	F	F
Système de fichiers	9.4.5	O		F	F	F
Temporisateur et horloge	9.4.6	O		F	F	F
Gestion de la consommation d'énergie	9.4.7	O		F	F	F
Paramètres de pays et de langue	9.4.8	O		F	F	F
API AS générale	9.5.2.2	O		O	O	O
API AS de déchiffrement	9.5.2.3	O		O	s. o.	O
API AS d'exportation	9.5.2.4	C	Pour enregistrement ou passerelle	F	s. o.	F
API AS de chiffrement	9.5.2.5	C	Pour enregistrement ou passerelle	s. o.	O	s. o.
Carte à puce	9.5.3	C	Si lecteur de carte à puce pris en charge	F	F	F
Acquisition de carrousel de données	9.5.4	C	Pour les réseaux de radiodiffusion	F	F	F
Déchiffrement (voir la NOTE)	9.6.2	O		O	s. o.	O
Connexion d'exportation	9.7.2.3	C	Pour enregistrement ou passerelle	F	s. o.	F
Connexion d'importation	9.7.2.4	C	Pour enregistrement ou passerelle	s. o.	O	s. o.

API	Paragraphe	Hôte	Condition (le cas échéant)	Client déchiff.	Micro serveur	Micro client
Rechiffrement (voir la NOTE)	9.7.2.5	C	Pour enregistrement ou passerelle	s. o.	O	s. o.
Redéchiffrement par le Micro client	9.7.2.6	O		F	s. o.	O
Paramètres de pays et de langue	9.4.8	O		F	F	F
Informations URI standard	9.8.2.3	O		O	O	O
Informations URI client	9.8.2.4	O		O	O	O
Informations URI de base	9.8.2.5	O		O	O	O
Contrôle de sortie	9.8.2.6	O		O	O	O
Insertion de filigranes	9.8.2.7	C	Pour les dispositifs dotés de capacités de radiodiffusion ou de multidiffusion	F	s. o.	F
Contrôle parental	9.8.2.8	O		O/F	O/F	O/F
Synchronisation des propriétés de contenu	9.8.2.9	O		O	O	O
Authentification parentale	9.8.2.10	O		F	s. o.	F
Délégation de l'authentification parentale	9.8.2.11	O		F	s. o.	F
API de communication entre clients	9.9.2	O		F	F	F
NOTE – Les créneaux peuvent être conçus spécifiquement pour les **Micro serveurs** et les clients de déchiffrement. Le créneau en lui-même est identique sur le plan technique, mais les ressources de **sécurité évoluée** requises et les fonctions de déchiffrement associées sont distinctes.						

L'API de découverte ne propose pas de mécanisme permettant à un **Hôte ECI** de détecter qu'un **Client ECI** peut déchiffrer ou chiffrer des données de média au format fichier et/ou TS (flux de transport). Cette capacité est indiquée par le champ mhType du paramètre decryptId dans le message setDcrMhMatch (voir § 9.6.2.2.2). En ce qui concerne le rechiffrement, cette découverte est assurée par le paramètre EciEncrModes du message setEncrModes (voir § 9.7.2.5.2).

- Un dispositif conforme **ECI** destiné uniquement à la consommation de contenu fournira au moins deux **Instances de Machine virtuelle** et deux **Créneaux AS**.
- Les **Hôtes ECI** qui prennent en charge la fonctionnalité d'enregistreur vidéo personnel prendront en charge au moins un conteneur (**Instance de Machine virtuelle**) et un **Créneau AS** supplémentaires pour un **Micro serveur**. Si ces **Hôtes ECI** proposent également une fonctionnalité de lecture du contenu enregistré, ils prendront en charge au moins un conteneur (**Instance de Machine virtuelle**) et un **Créneau AS** supplémentaires pour un **Micro client** pouvant décoder le contenu rechiffré.
- Les **Hôtes ECI** qui prennent en charge la fonctionnalité de passerelle de réseau prendront en charge au moins un conteneur (**Instance de Machine virtuelle**) et un **Créneau AS** supplémentaires pour un **Micro serveur**.

Annexe A

Fonctions cryptographiques de l'Hôte ECI

(Cette annexe fait partie intégrante de la présente Recommandation.)

A.1 Fonction de hachage

Les fonctions de hachage décrites dans la présente Recommandation sont toutes basées sur l'algorithme SHA-256 défini dans le document [NIST FIPS 197].

La fonction *hash* du § 5.2 correspond à la fonction SHA-256() définie dans le document [NIST FIPS 197].

La fonction C asHash(uchar *data, uint datalength, resultLength, uchar *result) utilise les octets commençant au niveau des données de longueur dataLength en tant que chaîne d'octets *dataIn*, calcule la chaîne d'octets resultOut comme resultLength/8, et stocke le résultat conformément à la fonction:

resultOut = BS2OSP(truncate(SHA-256(OS2BSP(*dataIn*)),resultLength)))

resultLength sera un multiple de 8. La fonction "truncate" sera la troncature par la gauche d'une chaîne de bits (paramètre 1) selon les bits de longueur (paramètre 2).

BS2OSP et OS2BSP sont des fonctions convertissant une chaîne de bits en chaîne d'octets et inversement, comme défini au § 9 de la Recommandation [UIT-T J.1014].

A.2 Chiffrement asymétrique

Les opérations de chiffrement et de déchiffrement asymétriques sont définies au § 12.4 de la Recommandation [UIT-T J.1014].

A.3 Chiffrement symétrique

Le chiffrement AES décrit dans la présente Recommandation se définira conformément au document [NIST FIPS 197], sauf si une référence spécifique d'application AES est fournie.

Les applications CBC du chiffrement AES seront telles que définies dans le document [NIST Block 2001], sauf si une référence spécifique relative au mode AES-CBC est fournie. Sauf définition contraire, le vecteur d'initialisation 0 sera utilisé.

Les applications CTR du chiffrement AES seront telles que définies dans le document [NIST Block 2001], sauf si une référence spécifique relative au mode AES-CTR est fournie. Sauf définition contraire, le vecteur d'initialisation 0 sera utilisé.

A.4 Génération de nombres aléatoires

La génération de nombres aléatoires décrite dans la présente Recommandation sera conforme à la spécification définie à l'Annexe A de la Recommandation [UIT-T J.1014].

Annexe B

Paramètres d'interopérabilité

(Cette annexe fait partie intégrante de la présente Recommandation.)

B.1 Introduction

Cette Annexe définit les paramètres liés aux exigences en matière de ressources dans les **Équipements CPE**. La conformité à ces exigences sert l'interopérabilité entre les **Clients ECI**, les services de sécurité **ECI** fournis par les réseaux, et les **Équipements CPE**.

B.2 Longueur de la Liste de révocation

Les **Équipements CPE** réserveront suffisamment de stockage non volatile pour stocker les **Listes de révocation**, aux longueurs ci-dessous, pour chaque élément pouvant être révoqué, tel que défini dans le Tableau B.2–1. L'**Autorité de confiance ECI** doit s'assurer que les **Listes de révocation** émises respectent ces limites.

Tableau B.2-1 – Longueur maximale des Listes de révocation

Liste de révocation	Nombre maximal d'identificateurs
Fabricant	500
Hôte	500
Fournisseur	500
Client ECI	500
opérateur	500
Opération de plate-forme	500

B.3 Taille de l'image de Client ECI

Un **Hôte ECI** doit disposer d'au moins 500 ko de stockage pour l'**Image de client ECI** par créneau de **Client ECI** qu'il prend en charge.

B.4 Paramètres de configuration du carrousel de radiodiffusion

L'interface **ECI** définit des temps d'acquisition maximaux tCdownloadScenario pour tous les éléments devant être téléchargés depuis un carrousel de radiodiffusion afin de garantir une bonne conception des **Hôtes ECI**. Le paramètre tCdownloadScenario reflète le temps de téléchargement réel; par conséquent, le taux de répétition du carrousel doit représenter au moins trois fois cette valeur afin de garantir le téléchargement par l'**Hôte ECI** dans cette limite. Les radiodiffuseurs doivent fournir une bande passante suffisante pour prendre en charge le taux de répétition requis.

L'interface **ECI** définit également une taille maximale de module aux fins d'attribution de la mémoire tampon.

Le paramètre tCdownloadScenario et la taille maximale de module que la conception de l'**Hôte ECI** doit lui permettre de prendre en charge sont définis dans le Tableau B.4-1.

Tableau B.4-1 – Durée maximale des scénarios de téléchargement et taille maximale de module pour les carrousels ECI

Type de table	tCdownloadScenario	Taille max. de module
images de Client ECI	5 minutes	500 ko
Données de révocation du **Client ECI**	5 minutes	100 ko par seau
Chaîne de certificats d'**Opération de plate-forme**	10 secondes	50 ko
Données de révocation d'**Opération de plate-forme**	5 minutes	100 ko par seau
Données de révocation d'**Hôte ECI**	5 minutes	100 ko par seau
Données de configuration de **sécurité évoluée**	2 minutes	20 ko par seau

Annexe C

Vue d'ensemble des API de l'Hôte ECI

(Cette annexe fait partie intégrante de la présente Recommandation.)

Le Tableau C.1 définit les valeurs de **MsgApiTag** comme défini au § 9.3.1.

Tableau C.1 – Schéma de numérotation des API ECI

API	Paragraphe	Valeur de MsgApiTag	Version la plus récente de l'API	Versions obsolètes de l'API
Découverte des interfaces de l'Hôte	9.4.2	0x0001	0x0000	Aucune
Interface homme-machine	9.4.3	0x0002	0x0000	Aucune
IP	9.4.4	0x0003	0x0000	Aucune
HTTP(S)	9.4.4.6	0x0004	0x0000	Aucune
Système de fichiers	9.4.5	0x0005	0x0000	Aucune
Temporisateur et horloge	9.4.6	0x0006	0x0000	Aucune
Gestion de la consommation d'énergie	9.4.7	0x0007	0x0000	Aucune
Paramètres de pays et de langue	9.4.8	0x0008	0x0000	Aucune
API AS générale	9.5.2.2	0x0009	0x0000	Aucune
API AS de déchiffrement	9.5.2.3	0x000A	0x0000	Aucune
API AS d'exportation	9.5.2.4	0x000B	0x0000	Aucune
API AS de chiffrement	9.5.2.5	0x000C	0x0000	Aucune
Carte à puce	9.5.3	0x000D	0x0000	Aucune
Acquisition de carrousel de données	9.5.4	0x000E	0x0000	Aucune
Déchiffrement	9.6.2	0x000F	0x0000	Aucune
Connexion d'exportation	9.7.2.3	0x0010	0x0000	Aucune
Connexion d'importation	9.7.2.4	0x0011	0x0000	Aucune
Rechiffrement	9.7.2.5	0x0012	0x0000	Aucune
Redéchiffrement par le Micro client	9.7.2.6	0x0013	0x0000	Aucune
Informations URI standard	9.8.2.3	0x0014	0x0000	Aucune
Informations URI client	9.8.2.4	0x0015	0x0000	Aucune
Informations URI de base	9.8.2.5	0x0016	0x0000	Aucune
Contrôle de sortie	9.8.2.6	0x0017	0x0000	Aucune
Insertion de filigranes	9.8.2.7	0x0018	0x0000	Aucune
Contrôle parental	9.8.2.8	0x0019	0x0000	Aucune
Synchronisation des propriétés de contenu	9.8.2.9	0x0020	0x0000	Aucune
Authentification parentale	9.8.2.10	0x0021	0x0000	Aucune
Délégation de l'authentification parentale	9.8.2.11	0x0022	0x0000	Aucune
API de communication entre clients	9.9.2	0x0023	0x0000	Aucune

Annexe D

Compatibilité en aval des définitions des propriétés de contenu

(Cette annexe fait partie intégrante de la présente Recommandation.)

Les propriétés de contenu doivent être implémentées de manière extrêmement fiable au moyen de matériel ou de micrologiciels de bas niveau, et peuvent être complexes, coûteuses ou impossibles à changer ou à mettre à jour après la production du système sur puce. Ce paragraphe explique l'approche adoptée pour permettre l'évolution de ces propriétés malgré les limitations en termes de mise à niveau.

De nouvelles propriétés de contenu et/ou un développement des fonctionnalités liées aux propriétés de contenu existantes pourront s'avérer nécessaires à un stade ultérieur. Il peut être question également de l'augmentation du nombre de bits représentant la valeur d'une propriété de contenu. L'implémentation des propriétés de contenu dans un ancien **Hôte ECI** ne connaît pas les nouvelles fonctionnalités et il est souvent impossible de procéder à une mise à jour. Les propriétés de contenu sur les **Hôtes ECI** sont définies de façon à permettre une compatibilité en aval maximale avec les nouvelles fonctionnalités.

Les **Hôtes ECI** ont un comportement défini pour toutes les valeurs d'entrée et ignoreront toutes les extensions de champ pour lesquelles ils ne sont pas conçus. Ils créent en outre un comportement défini, c'est-à-dire que chaque valeur d'une propriété de contenu future aura un *comportement défini unique* sur tous les **Hôtes ECI** n'implémentant pas toutes les extensions, y compris les **Hôtes ECI** appliquant la première version de la propriété de contenu. Grâce à ce principe, de nouvelles valeurs de propriété de contenu peuvent être attribuées tout en sachant quel comportement en résultera sur les versions précédentes des implémentations de l'**Hôte ECI**. Si une nouvelle propriété de contenu doit avoir deux options différentes (ou plus) pour l'interprétation rétrocompatible par les anciens **Hôtes ECI**, deux valeurs réservées (ou plus) peuvent être attribuées, en étant dotées de la même sémantique dans la définition de la nouvelle propriété de contenu mais en ayant chacune une interprétation rétrocompatible appropriée (quoique différente).

Prenons comme exemple de champ d'extension un nouveau champ de contrôle de sortie devant être défini pour un nouveau type de sortie X dans l'API de contrôle de sortie. On l'attribue au bit 5, qui est réservé dans la version 1. Il peut utiliser la même sémantique que le champ OcIP. Toute implémentation antérieure des **Clients ECI** attribuera à ce champ la valeur 0. L'interprétation par un ancien **Hôte ECI** sera la suivante:

- si OcAnyOther==0b0, la sortie X est autorisée;
- si OcAnyOther==0b1, la sortie X est interdite.

Ceci correspond tout à fait à la sémantique dans une nouvelle implémentation de l'**Hôte ECI** lorsque OcX==0b0. Toutefois, lorsque OcX==0b1, l'autorisation de la sortie sera inversée par rapport à l'ancienne configuration avec OcX==0b0, permettant ainsi de nouvelles fonctionnalités dans la combinaison entre un nouvel **Hôte ECI** et un nouveau **Client ECI**. Il convient de noter que l'interprétation inversée des valeurs de champ en fonction d'OcAnyOther garantit que la valeur 0 attribuée à un champ non défini prend sa valeur naturelle: autorisation maximale pour OcAnyOther==0b0 (autres sorties autorisées) et autorisation minimale pour OcAnyOther==0b1 (autres sorties interdites).

Inversement, il est important que les **Clients ECI** n'utilisant pas la dernière définition de la propriété de contenu ne fassent pas référence à une nouvelle fonctionnalité de cette définition dont ils n'ont pas connaissance, ou a fortiori, qu'ils n'utilisent pas ces valeurs prétendument non attribuées à titre privé, car ces valeurs correspondent à des comportements définis dans tous les **Hôtes ECI**. Une telle utilisation inappropriée créera un sérieux obstacle à l'intégration de ces valeurs pour des

fonctionnalités définies par l'interface **ECI**. C'est la raison pour laquelle cette spécification interdit explicitement l'application par les **Clients ECI** de valeurs de propriété de contenu non attribuées.

Plus particulièrement, pour les champs pouvant avoir plusieurs valeurs, les valeurs réservées correspondront toutes à un comportement défini sur les **Hôtes ECI**, mais elles ne doivent pas être utilisées par les **Clients ECI**.

Tout sous-champ non attribué dans une définition de propriété de contenu doit avoir un comportement défini sur un **Hôte ECI** correspondant à l'une des valeurs définies de la propriété de contenu. De manière générale, un **Hôte ECI** ignorera ces sous-champs, c'est-à-dire qu'il interprétera la valeur de la propriété de contenu uniquement en fonction des champs définis. Les **Clients ECI** attribueront généralement la valeur 0 à ces sous-champs. Toute divergence de cette politique d'attribution de la valeur 0 à un champ non attribué doit être prédéfinie par une version de la définition de la propriété de contenu.

Toute extension de champ sera ignorée par les **Hôtes ECI** appliquant la définition de propriété de contenu correspondante et les **Clients ECI** attribuant les valeurs assigneront la valeur 0 à ces extensions de champ.

Appendice I

Liste de tous les messages d'API disponibles dans l'ordre alphabétique

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

Les messages d'API répertoriés dans l'Appendice I sont extraits des tableaux suivants du § 9 de la présente Recommandation (Tableau I.1).

Tableau I.1 – Liste des tableaux répertoriant les messages des différentes API

API	Paragraphe	Catégorie d'API
API de découverte des interfaces de l'Hôte	9.4.2.1-1	
API d'interfaces d'utilisateur	9.4.3.1-1	
API des sockets IP	9.4.4.3.1-1	
API des sockets UDP	9.4.4.4.1-1	
API des sockets TCP	9.4.4.5.1-1	
API HTTP GET	9.4.4.6.1-1	
API d'ouverture et de fermeture des fichiers	9.4.5.2.1-1	
API d'accès aux fichiers	9.4.5.3.1-1	
API des services de répertoire de fichiers	9.4.5.4.1-1	API générales
API du temporisateur	9.4.6.2.1-1	
API d'horloge	9.4.6.3.1-1	
API de gestion des transitions de consommation d'énergie	9.4.7.2-1	
API de sortie du mode veille	9.4.7.3-1	
API de définition des paramètres de pays et de langue	9.4.8.1-1	
API AS générale	9.5.2.2.1-1	
API AS de déchiffrement	9.5.2.3.1-1	
API AS d'exportation	9.5.2.4.1-1	
API AS de chiffrement	9.5.2.5.1-1	API spécifiques à l'interface **ECI**
API de gestion des sessions de **Carte à puce**	9.5.3.6.1–1	
API de communication avec la **Carte à puce**	9.5.3.6.1–1	
API d'acquisition de carrousel de données	9.5.4.1-1	
API de session de déchiffrement d'un Pointeur de média	9.6.2.2.1-1	
API de Connexion d'exportation	9.7.2.3.1-1	
API de Connexion d'importation	9.7.2.4.1-1	
API de rechiffrement	9.7.2.5.1-1	
API de déchiffrement	9.7.2.6.1-1	
API relative aux droits d'utilisation et au contrôle parental	9.8.2.1-1	
API de communication entre clients	9.9.2.1-1	

Le Tableau I.2 répertorie tous les messages d'API dans l'ordre alphabétique.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
1	callAsNextKeySession	API AS générale	9.5.2.2.3	S	C→H	Passer à la clé aléatoire suivante pour une session.
2	callCardGetProp	**Carte à puce**	9.5.3.6.5	S	H→C	Obtenir la propriété/le paramètre de communication avec la carte.
3	callCardSessionPrio	**Carte à puce**	9.5.3.5.3	S	C→H	Définir la priorité de la session de **Carte à puce**.
4	callCardSetProp	**Carte à puce**	9.5.3.6.4	S	H→C	Définir le paramètre de communication avec la carte.
5	callFileDataLog	Système de fichiers	9.4.5.3.6	S	C→H	Ajoute des données à la fin d'un fichier placé dans la mémoire tampon.
6	callLocaltime	Horloge	9.4.6.3.3	S	C→H	Convertit le nombre entier en heure locale.
7	getApis	Découverte des interfaces	9.4.2.2	S	C→H	Lit les API de l'Hôte disponibles.
8	getApiVersions	Découverte des interfaces	9.4.2.3	S	C→H	Lit les versions disponibles d'une API de l'Hôte.
9	getAsClientRnd	API AS générale	9.5.2.2.13	S	C→H	Obtenir un nouveau nombre aléatoire pour les applications du **Client ECI**.
10	getAsSC	API AS générale	9.5.2.2.14	S	C→H	Obtenir le statut du champ de commande d'embrouillage actuel du contenu d'une session.
11	getAsSessionLimitCounter	API AS générale	9.5.2.2.10	S	C→H	Obtenir la valeur du compteur de limite pour la session.
12	getAsSessionRk	API AS générale	9.5.2.2.9	S	C→H	Obtenir une valeur de clé aléatoire pour une session.
13	getAsSlotRk	API AS générale	9.5.2.2.8	S	C→H	Obtenir une valeur de clé aléatoire pour le **Créneau AS**.
14	getCardConnStatus	**Carte à puce**	9.5.3.5.4	S	H→C	Fournit le statut de connexion de la carte.
15	getChipsetId	API AS générale	9.5.2.2.16	S	C→H	Obtenir la valeur ChipsetID du **Bloc d'échelle de clés**
16	getDcrMarkMeta	Propriétés de contenu	9.8.2.7.4	S	H→C	Lire une propriété du système d'insertion de filigranes.
17	getDcrMarkSyst	Propriétés de contenu	9.8.2.7.2	S	H→C	Obtenir les systèmes d'insertion de filigranes pris en charge.
18	getDcrTsSource	Contrôle de la source du flux de transport de déchiffrement	9.6.2.3.6.2	S	C→H	Le **Client ECI** obtient la source du flux de transport.
19	getEncrStdUri	Propriétés de contenu	9.8.2.3.2	S	C→H	Obtenir les informations URI standard pour le contenu à rechiffrer.
20	getEncrBasicUri	Propriétés de contenu	9.8.2.5.2	S	C→H	Obtenir les informations URI de base pour le contenu à rechiffrer.
21	getEncrCustUri	Propriétés de contenu	9.8.2.4.2	S	C→H	Obtenir des informations URI personnalisées pour le contenu à rechiffrer.
22	getEncrOutputCtrl	Propriétés de contenu	9.8.2.6.2	S	C→H	Obtenir les restrictions relatives au contrôle de sortie pour le contenu à rechiffrer.
23	getEncrParCtrl	Propriétés de contenu	9.8.2.8.2	S	C→H	Obtenir les conditions relatives au contrôle parental pour le contenu à désembrouiller.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
24	getIccClientInfo	Communication entre clients	9.9.2.4	S	C→H	Le **Client ECI** lit l'identité et le statut de connexion d'un autre **Client ECI** dans le système.
25	getIccMaxClients	Communication entre clients	9.9.2.2	S	C→H	Le **Client ECI** lit le nombre maximal de **Clients ECI** que l'**Hôte ECI** peut prendre en charge.
26	getImageTargetId	API AS générale	9.5.2.2.17	S	C→H	Obtenir la valeur ECI_Image_Target_Id de l'équipement CPE
27	getPwrStatus	Gestion de la consommation d'énergie	9.4.7.2.2	S	C→H	Obtient la valeur actuelle de l'état de consommation d'énergie.
28	getTime	Horloge	9.4.6.3.2	S	C→H	Lit l'horloge système locale sous forme de nombre entier.
29	reqAsAStartDecryptSession	API AS de déchiffrement	9.5.2.3.2	A	C→H	Démarrer une session de déchiffrement dans le **Créneau AS** du **Client ECI**.
30	reqAsAuthDecrSlotConfig	API AS de déchiffrement	9.5.2.3.4	A	H→C	Authentifier la configuration du créneau au moyen de mécanismes d'authentification (mode de déchiffrement).
31	reqAsAuthEncrSlotConfig	API AS de chiffrement	9.5.2.5.5	A	C→H	Authentifier la configuration du créneau et les paramètres de chiffrement au moyen de mécanismes d'authentification (mode de chiffrement).
32	reqAsClientChalResp	API AS générale	9.5.2.2.7	A	C→H	Appliquer la clé d'authentification du **Client ECI** aux données et renvoyer un résultat.
33	reqAsComputeAkClient	API AS générale	9.5.2.2.6	A	C→H	Calculer la clé d'authentification pour les applications du **Client ECI**.
34	reqAsComputeEncrCw	API AS de chiffrement	9.5.2.5.4	A	C→H	Calculer un mot de contrôle de chiffrement.
35	reqAsEventCpChange	API AS de chiffrement	9.5.2.5.8	A	H→C	Message d'événement signalant un changement dans les propriétés du contenu importé dans une session de chiffrement.
36	reqAsEventSC	API AS générale	9.5.2.2.15	A	H→C	Message d'événement lors du changement du champ de commande d'embrouillage dans la session.
37	reqAsEventSessionLimit	API AS générale	9.5.2.2.12	A	H→C	Lorsqu'une valeur limite est atteinte pour les unités restantes, envoyer l'événement au **Client ECI**.
38	reqAsExportConnEnd	API AS d'exportation	9.5.2.4.3	A	C→H	Terminer la session d'exportation existante.
39	reqAsExportConnSetup	API AS d'exportation	9.5.2.4.2	A	C→H	Configurer une **Connexion d'exportation** entre des sessions de déchiffrement et de chiffrement.
40	reqAsInitSlot	API AS générale	9.5.2.2.2	A	C→H	Initialise le **Créneau AS**.
41	reqAsLdUssk	API AS de chiffrement	9.5.2.5.6	A	C→H	Charger la clé secrète du **Micro serveur**.
42	reqAsLoadSlotLk	API AS générale	9.5.2.2.5	A	C→H	Calculer la clé de liaison de niveau supérieur (LK1).
43	reqAsMInikLk1	API AS de chiffrement	9.5.2.5.7	A	C→H	Calculer le message d'initialisation asymétrique du **Micro client**.
44	reqAsStartEncryptSession	API AS de chiffrement	9.5.2.5.3	A	C→H	Démarrer une session de chiffrement.
45	reqAsStopSession	API AS générale	9.5.2.2.4	A	C→H	Arrêter une session.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
46	reqCardCmdRes	**Carte à puce**	9.5.3.6.2	A	C→H	Envoyer une commande à la carte, recevoir une réponse de la carte.
47	reqCardReInit	**Carte à puce**	9.5.3.6.3	A	C→H	Effectuer une remise à zéro (à chaud ou à froid) de la carte et réexécuter la séquence d'initialisation avec le dernier paramètre de préférence d'initialisation.
48	reqCCardConClose	**Carte à puce**	9.5.3.5.6	A	H→C	Informe le **Client ECI** qu'une session de carte a été fermée.
49	reqCCardConOpen	**Carte à puce**	9.5.3.5.5	A	H→C	Informe le **Client ECI** qu'une session de carte a été ouverte.
50	reqCCountry	Pays	9.4.8.2.2	A	H→C	L'**Hôte ECI** demande la préférence de pays existante du **Client ECI**.
51	reqCLanguage	Langue	9.4.8.2.4	A	H→C	L'**Hôte ECI** demande la préférence de langue existante du **Client ECI**.
52	reqCpChange	Propriétés de contenu	9.8.2.9.2	A	H→C	L'**Hôte ECI** signale un changement à venir dans les propriétés du contenu devant être rechiffré.
53	reqDCAcqModule	Acquisition de carrousel de données	9.5.4.3	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** d'acquérir un module de carrousel de données **ECI** particulier dans un fichier au moyen de paramètres de filtre de module et de divers modes.
54	reqDCAcqGroupInfo	Acquisition de carrousel de données	9.5.4.2	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** de lire la structure GroupInfoIndication dans le message DSI du carrousel de données **ECI** spécifié.
55	reqDcrFileQuit	Déchiffrement de fichier média	9.6.2.4.4.4	A	C→H	Le **Client ECI** annule une session de désembrouillage avec l'**Hôte ECI**.
56	reqDcrFileData	Demande de données par le biais d'un filtre de fichiers	9.6.2.4.5.2.4	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** d'acquérir des données par l'intermédiaire du filtre de fichiers.
57	reqDcrFileStop	Déchiffrement de fichier média	9.6.2.4.4.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'arrêter le désembrouillage au niveau d'un **Pointeur de média**.
58	reqDcrFileFilter	Demande de filtre de fichiers	9.6.2.4.5.2.3	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** de définir un filtre de données pour l'acquisition de données de sécurité.
59	reqDcrFileKeyComp	Demande de calcul d'une clé	9.6.2.4.6.3	A	H→C	Initier toute activité de calcul ou autre de la part du **Client ECI** en vue de mettre à disposition un mot de contrôle pourvu d'un identificateur de clé.
60	reqDcrFileStart	Déchiffrement de fichier média	9.6.2.4.4.2	A	H→C	Demande au **Client ECI** de désembrouiller un fichier ou un flux ou de renvoyer son statut de désembrouillage.
61	reqDcrIpServer	Rechiffrement	9.7.2.6.5	A	C→H	Le **Micro client** demande à l'**Hôte ECI** de fournir l'adresse IP du **Micro serveur** en vue des communications ultérieures liées à la session de **Pointeur de média**.
62	reqDcrMhBcAlloc	Déchiffrement de pointeur de média	9.6.2.2.5	A	C→H	Le **Client ECI** demande un **Pointeur de média** pour ses propres besoins d'accès à un réseau de radiodiffusion.
63	reqDcrMhCancel	Déchiffrement de pointeur de média	9.6.2.2.6	A	C→H	Le **Client ECI** annule une session de média avec l'**Hôte ECI**.
64	reqDcrMhClose	Déchiffrement de pointeur de média	9.6.2.2.4	A	H→C	L'**Hôte ECI** ferme une session de média avec un **Client ECI**.
65	reqDcrMhOpen	Déchiffrement de pointeur de média	9.6.2.2.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'ouvrir une session de média d'un type

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
						spécifié au moyen d'un **Pointeur de média**.
66	reqDcrMsgRecv	Rechiffrement	9.7.2.6.7	A	H→C	L'**Hôte ECI** fournit au **Micro client** un message issu du **Micro serveur** d'une session de **Pointeur de média**.
67	reqDcrMsgSend	Rechiffrement	9.7.2.6.6	A	C→H	Le **Micro client** demande à l'**Hôte ECI** d'envoyer un message au **Micro serveur** d'une session de **Pointeur de média**.
68	reqDcrTargetCred	Rechiffrement	9.7.2.6.4	A	H→C	L'**Hôte ECI** demande au **Client ECI** de fournir les données d'initialisation d'une connexion avec le **Micro serveur** généralement utilisée pour l'authentification de la **Cible**.
69	reqDcrTargets	Rechiffrement	9.7.2.6.3	A	H→C	L'**Hôte ECI** demande au **Micro client** de fournir les cibles de chiffrement pour lesquelles il peut déchiffrer des services.
70	reqDcrTsData	Rechiffrement	9.7.2.6.8	A	C→H	Le **Micro serveur** fournit à l'**Hôte ECI** les données devant être transférées au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'ECM.
71	reqDcrTsDescrquit	Déchiffrement du contenu du flux de transport	9.6.2.3.4.4	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** de mettre fin au désembrouillage d'une session de **Pointeur de média**.
72	reqDcrTsData	Redéchiffrement par le Micro client	6.7.2.6.7	A	H→C	L'**Hôte ECI** fournit au **Micro client** les données requises (très) prochainement pour le déchiffrement du contenu sur le **Pointeur de média**.
73	reqDcrTsDescrStop	Déchiffrement du contenu du flux de transport	9.6.2.3.4.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'arrêter le désembrouillage d'une session de **Pointeur de média**.
75	reqDcrTsDescrStart	Déchiffrement du contenu du flux de transport	9.6.2.3.4.2	A	H→C	Demande au **Client ECI** de désembrouiller un programme dans un flux de transport ou de renvoyer son statut de désembrouillage.
76	reqDcrTsRelocate	Contrôle de la source du flux de transport de déchiffrement	9.6.2.3.6.3	A	C→H	Le **Client ECI** déplace la source du flux de transport.
77	reqDcrTsSection	Acquisition des données du flux de transport de déchiffrement	9.6.2.3.5.5	A	H→C	Transfert une section acquise au **Client ECI**.
78	reqDcrTsSelectCancel	Contrôle de la source du flux de transport de déchiffrement	9.6.2.3.6.6	A	C→H	Le **Client ECI** annule sa sélection de programme précédente.
79	reqDcrTsSelectPmt	Contrôle de la source du flux de transport de déchiffrement	9.6.2.3.6.5	A	C→H	Le **Client ECI** sélectionne un programme dans le flux de transport en fonction de la PMT.
80	reqDcrTsSelectPrg	Contrôle de la source du flux de transport de déchiffrement	9.6.2.3.6.4	A	C→H	Le **Client ECI** sélectionne un programme dans le flux de transport en fonction du numéro de programme.
81	reqDcrTsTable	Acquisition des données du flux de transport de déchiffrement	9.6.2.3.5.6	A	C→H	Le **Client ECI** acquiert une table dans le flux.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
82	reqEncrConnDrop	Rechiffrement	9.7.2.5.5	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'abandonner toute information relative à une connexion de rechiffrement précédemment pré-authentifiée.
83	reqEncrConnSetup	Rechiffrement	9.7.2.5.4	A	H→C	L'**Hôte ECI** demande au **Client ECI** de créer une connexion avec la **Cible** du rechiffrement et de pré-authentifier la **Cible** du rechiffrement pour référence ultérieure lors de la configuration d'une session de **Pointeur de média**.
84	reqEncrFileData	Rechiffrement	9.7.2.5.18	A	C→H	Le **Micro serveur** fournit à l'**Hôte ECI** un message devant être transféré au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'identificateur de clé.
85	reqEncrIpServer	Rechiffrement	9.7.2.5.13	A	H→C	L'**Hôte ECI** demande l'adresse de serveur IP d'un **Micro serveur** afin de permettre aux **Micro clients** de créer des connexions IP.
86	reqEncrMhCancel	Rechiffrement	9.7.2.5.9	A	C→H	Le **Client ECI** met fin à la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.
87	reqEncrMhClose	Rechiffrement	9.7.2.5.8	A	H→C	L'**Hôte ECI** ferme la **Session de rechiffrement** avec le **Client ECI**.
88	reqEncrMhOpen	Rechiffrement	9.7.2.5.7	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'ouvrir une session de **Pointeur de média** afin de rechiffrer le contenu issu d'une **Connexion d'importation** entrante à destination d'une connexion de rechiffrement établie.
89	reqEncrMhQuit	Rechiffrement	9.7.2.5.12	A	C→H	Le **Client ECI** informe l'**Hôte ECI** que l'opération de rechiffrement sur le **Pointeur de média** a pris fin.
90	reqEncrMhStart	Rechiffrement	9.7.2.5.10	A	H→C	L'**Hôte ECI** demande au **Client ECI** de démarrer l'opération de rechiffrement pour une session de **Pointeur de média**.
91	reqEncrMhStop	Rechiffrement	9.7.2.5.11	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'arrêter une opération de rechiffrement pour une session de **Pointeur de média**.
92	reqEncrMsgRecv	Rechiffrement	9.7.2.5.18	A	H→C	L'**Hôte ECI** fournit au **Micro serveur** un message issu de la **Cible** d'une session de **Pointeur de média**.
93	reqEncrMsgSend	Rechiffrement	9.7.2.5.14	A	C→H	Le **Micro serveur** demande à l'**Hôte ECI** de transférer un message à la **Cible** d'une session de **Pointeur de média**.
94	reqEncrTargets	Rechiffrement	9.7.2.5.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** de fournir les nœuds de **Cible** qu'il peut authentifier.
95	reqEncrTsData	Rechiffrement	9.7.2.5.16	A	C→H	Le **Micro serveur** fournit à l'**Hôte ECI** les données devant être transférées au **Micro client cible** d'un **Pointeur de média** pour déchiffrement, y compris les informations de synchronisation liées à l'ECM.
96	reqEncrTsEcm	Rechiffrement	9.7.2.5.17	A	C→H	Le **Micro serveur** émet une section ECM requise par le **Micro client** pour le déchiffrement au cours de la période cryptographique suivante.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
97	reqExpConnCancel	Connexion d'exportation	9.7.2.3.5	A	C→H	Le **Client ECI** met fin à une **Connexion d'exportation** initialisée avec un **Client ECI** importateur.
98	reqExpConnDrop	Connexion d'exportation	9.7.2.3.4	A	H→C	L'**Hôte ECI** annule toute connexion initialisée précédemment entre un **Client ECI** exportateur et un **Client ECI** importateur.
99	reqExpConnNodes	Connexion d'exportation	9.7.2.3.2	A	H→C	L'**Hôte ECI** demande les nœuds des options d'exportation au **Client ECI**.
100	reqExpConnSetup	Connexion d'exportation	9.7.2.3.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** d'initialiser une **Connexion d'exportation** vers un **Client ECI** importateur sur la base d'une **Chaîne d'importation**.
101	reqExpMhCancel	Connexion d'exportation	9.7.2.3.8	A	C→H	Le **Client ECI** annule une session d'exportation.
102	reqExpMhClose	Connexion d'exportation	9.7.2.3.7	A	H→C	L'**Hôte ECI** ferme une session d'exportation.
103	reqExpMhOpen	Connexion d'exportation	9.7.2.3.6	A	H→C	L'**Hôte ECI** demande au **Client ECI** de créer une session d'exportation à partir d'une **Connexion d'exportation** initialisée précédemment.
104	reqFileClose	Système de fichiers	9.4.5.2.3	A	C→H	Ferme un fichier ouvert.
105	reqFileCreate	Système de fichiers	9.4.5.4.3	A	C→H	Crée un nouveau fichier.
106	reqFileDelete	Système de fichiers	9.4.5.4.4	A	C→H	Supprime un fichier.
107	reqFileDir	Système de fichiers	9.4.5.4.5	A	C→H	Répertorie les noms des fichiers disponibles dans le système de fichiers des **Clients ECI**.
108	reqFileOpen	Système de fichiers	9.4.5.2.2	A	C→H	Ouvre un fichier privé d'un **Client ECI**.
109	reqFileRead	Système de fichiers	9.4.5.3.3	A	C→H	Lit des octets consécutifs à partir de l'emplacement existant dans le fichier.
110	reqFileRemoveData	Système de fichiers	9.4.5.3.5	A	C→H	Supprime des données d'un fichier à partir d'un emplacement existant.
111	reqFileSeek	Système de fichiers	9.4.5.3.4	A	C→H	Repositionne l'emplacement existant dans le fichier.
112	reqFileStat	Système de fichiers	9.4.5.4.2	A	C→H	Renvoie la taille et la date/heure de modification du fichier.
113	reqFileWrite	Système de fichiers	9.4.5.3.2	A	C→H	Écrit des octets consécutifs à partir de l'emplacement existant dans le fichier.
114	reqHCardConClose	**Carte à puce**	9.5.3.5.7	A	C→H	Informe l'**Hôte ECI** que le **Client ECI** souhaite terminer une session avec la carte connectée.
115	reqHCountry	Pays	9.4.8.2.1	A	C→H	Demande la préférence de pays existante de l'**Hôte ECI**.
116	reqHLanguage	Langue	9.4.8.2.3	A	C→H	Demande la préférence de langue existante de l'**Hôte ECI**.
117	reqHttpGetData	HTTP GET	9.4.4.6.3	A	C→H	Exécute une requête HTTP GET sur une adresse URL et transmet le résultat au Client sous forme de données.
118	reqHttpGetFile	HTTP GET	9.4.4.6.3	A	C→H	Exécute une requête HTTP GET sur une adresse URL et stocke le résultat dans un fichier.
119	reqIccPipeCancel	Communication entre clients	9.9.2.7	A	C→H	Le **Client ECI** annule le conduit.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
120	reqIccPipeClose	Communication entre clients	9.9.2.8	A	H→C	L'**Hôte ECI** informe le **Client ECI** que le conduit avec son homologue a été fermé.
121	reqIccPipeMsgRecv	Communication entre clients	9.9.2.10	A	H→C	Le **Client ECI** reçoit un message de la part de son homologue de conduit.
122	reqIccPipeMsgSend	Communication entre clients	9.9.2.9	A	C→H	Le **Client ECI** envoie un message à son homologue de conduit.
123	reqIccPipeOpen	Communication entre clients	9.9.2.5	A	C→H	Demande d'ouverture de conduit vers un autre **Client ECI**.
124	reqIccPipeOpenReq	Communication entre clients	9.9.2.6	A	H→C	Demande entrante d'ouverture de conduit de la part d'un autre **Client ECI**.
125	reqIccSystemReady	Communication entre clients	9.9.2.3	A	H→C	L'**Hôte ECI** informe le **Client ECI** que tous les **Clients ECI** sont initialisés.
126	reqImpConnCancel	Connexion d'importation	9.7.2.4.6	A	C→H	Le **Client ECI** met fin à la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.
127	reqImpConnChain	Connexion d'importation	9.7.2.4.3	A	H→C	L'**Hôte ECI** demande au **Client ECI** importateur de fournir la chaîne d'entrée d'un nœud d'importation spécifique.
128	reqImpConnChainRenew	Connexion d'importation	9.7.2.4.3	A	C→H	Le **Client ECI** demande à l'**Hôte ECI** de réinitialiser la connexion avec une **Chaîne d'importation** mise à jour.
129	reqImpConnDrop	Connexion d'importation	9.7.2.4.5	A	H→C	L'**Hôte ECI** abandonne la **Connexion d'importation** avec le **Client ECI** exportateur spécifié.
130	reqImpConnNodes	Connexion d'importation	9.7.2.4.2	A	H→C	L'**Hôte ECI** demande au **Client ECI** importateur de fournir ses nœuds d'importation.
131	reqImpConnSetup	Connexion d'importation	9.7.2.4.4	A	H→C	L'**Hôte ECI** demande au **Client ECI** importateur d'initialiser une **Connexion d'importation** avec un **Client ECI** exportateur spécifique par l'intermédiaire d'un nœud d'importation.
132	reqIpAddrinfo	Sockets IP	9.4.4.3.4	A	C→H	Obtient l'adresse d'un **Hôte ECI** (distant).
133	reqIpClose	Sockets IP	9.4.4.3.3	A	C→H	Ferme un socket IP **ECI**.
134	reqIpSocket	Sockets IP	9.4.4.3.2	A	C→H	Ouvre un socket IP **ECI**.
135	reqIpTcpAccept	Socket TCP/IP	9.4.4.5.5	A	C→H	Le serveur TCP homologue accepte la connexion en provenance du client TCP homologue.
136	reqIpTcpConnect	Socket TCP/IP	9.4.4.5.2	A	C→H	Le client TCP se connecte au serveur TCP homologue.
137	reqIpTcpRecv	Socket TCP/IP	9.4.4.5.4	A	C→H	Reçoit des données de l'homologue connecté.
138	reqIpTcpSend	Socket TCP/IP	9.4.4.5.3	A	C→H	Envoie des données à l'homologue connecté.
139	reqIpUdpRecvMsg	Socket UDP/IP	9.4.4.4.3	A	C→H	Reçoit un message du port UDP de l'homologue.
140	reqIpUdpSendMsg	Socket UDP/IP	9.4.4.4.2	A	C→H	Envoie un message au port UDP de l'homologue.
141	reqParAuthChk	Propriétés de contenu	9.8.2.10.3	A	C→H	Demande à l'**Hôte ECI** d'effectuer une authentification parentale pour le compte du **Client ECI**.
142	reqParAuthChkCan	Propriétés de contenu	9.8.2.10.4	A	C→H	Annule une requête précédente d'authentification parentale envoyée à l'Hôte.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
143	reqParAuthCid	Propriétés de contenu	9.8.2.10.5	A	H→C	Demande l'autorisation parentale par code PIN pour un (futur) élément de contenu à décoder. Ce message peut déclencher un dialogue d'authentification parentale.
144	reqParAuthDel	Propriétés de contenu	9.8.2.11.2	A	H→C	L'**Hôte ECI** délègue une authentification parentale à un **Client ECI**.
145	reqParAuthDelCan	Propriétés de contenu	9.8.2.11.3	A	H→C	L'**Hôte ECI** annule une requête précédente d'authentification parentale envoyée au **Client ECI**.
146	reqPwrChange	Gestion de la consommation d'énergie	9.4.7.2.4	A	H→C	Notification d'un changement d'état de consommation d'énergie.
147	reqTimerCancel	Temporisateur	9.4.6.2.3	A	C→H	Annule un événement de temporisation précédemment défini.
148	reqTimerEvent	Temporisateur	9.4.6.2.2	A	C→H	Définit un événement de temporisation futur.
149	reqUiClientQuery	Interface d'utilisateur	9.4.3.4.8	A	H→C	Le **Client ECI** reçoit une requête de l'application HTML dans le navigateur et fournit une réponse (dynamique).
150	reqUiContainerMount	Interface d'utilisateur	9.4.3.4.2	A	C→H	Monte un conteneur d'applications d'interfaces d'utilisateur à l'aide de ressources HTML afin de prendre en charge des sessions d'interface d'utilisateur.
151	reqUiSessionCancel	Interface d'utilisateur	9.4.3.4.7	A	H→C	L'**Hôte ECI** annule une session d'interface d'**Utilisateur**.
152	reqUiSessionClose	Interface d'utilisateur	9.4.3.4.6	A	C→H	Le **Client ECI** met fin à une session d'interface d'**Utilisateur**.
153	reqUiSessionCommence	Interface d'utilisateur	9.4.3.4.4	A	H→C	L'**Hôte ECI** suggère au **Client ECI** d'ouvrir une session d'interface d'utilisateur.
154	reqUiSessionOpen	Interface d'utilisateur	9.4.3.4.5	A	C→H	Le **Client ECI** demande l'ouverture d'une session d'interface d'**Utilisateur** avec l'**Utilisateur** et affiche un contenu.
155	reqPwrWakeupEvent	Gestion de la consommation d'énergie	9.4.7.3	A	H→C	Signale l'expiration de la temporisation de réveil.
156	setApiVersion	Découverte des interfaces	9.4.2.4	S	C→H	Écrit la version de l'API de l'Hôte à utiliser.
157	setAsPermitCPChange	API AS de chiffrement	9.5.2.4	S	C→H	Activer/désactiver le changement des propriétés du contenu importé influant sur la sélection des mots de contrôle en vue du chiffrement dans une session de chiffrement.
158	setAsSC	API AS de chiffrement	9.5.2.4	S	C→H	Définir le champ de commande d'embrouillage du contenu chiffré d'une session de chiffrement.
159	setAsSessionLimitEvent	API AS générale	9.5.2.5.11	S	C→H	Définir la valeur limite pour l'envoi d'un message reqAsEventSessionLimit au **Client ECI**.
160	setCardMatch	**Carte à puce**	9.5.3.5.2	S	C→H	Définir la liste de spécificateurs d'identificateur de carte pour le **Client ECI**.
161	setCpSync	Propriétés de contenu	9.8.2	S	C→H	Le **Client ECI** indique que l'ensemble actuel de propriétés de contenu est cohérent et peut être appliqué au contenu devant être désembrouillé par un mot de contrôle à venir.

Tableau I.2 – Liste de tous les messages d'API dans l'ordre alphabétique

N°	Message	API	Paragraphe	Type	Sens	Description
162	setDcrBasicUri	Propriétés de contenu	9.8.2.5.1	S	C→H	Définir les informations URI de base pour le contenu à désembrouiller.
163	setDcrCustUri	Propriétés de contenu	9.8.2.4.1	S	C→H	Définir des informations URI personnalisées pour le contenu à désembrouiller.
164	setDcrMarkBasic	Propriétés de contenu	9.8.2.7.5	S	C→H	Définir la charge utile de base du filigrane pour le contenu à désembrouiller.
165	setDcrMarkExt	Propriétés de contenu	9.8.2.7.6	S	C→H	Définir la charge utile étendue du filigrane pour le contenu à désembrouiller.
166	setDcrMarkMeta	Insertion de filigranes	9.8.2.7.3	S	C→H	Définir une valeur de contrôle pour le système d'insertion de filigranes.
167	setDcrMhMatch	Déchiffrement de pointeur de média	9.6.2.2.2	S	C→H	Indique à l'**Hôte ECI** sous quels identificateurs le **Client ECI** peut être reconnu pour le désembrouillage du contenu.
168	setDcrModes	Rechiffrement	9.7.2.6.1	S	C→H	Le **Micro client** informe l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.
170	setDcrOutputCtl	Propriétés de contenu	9.8.2.6.1	S	C→H	Définir les restrictions relatives au contrôle de sortie pour le contenu à désembrouiller.
171	setDcrParCtl	Propriétés de contenu	9.8.2.8.1	S	C→H	Définir les conditions relatives au contrôle parental pour le contenu à désembrouiller.
172	setDcrStdUri	Propriétés de contenu	9.8.2.8.1	S	C→H	Définir les informations URI standard pour le contenu à désembrouiller.
173	setDcrTsSectionAcq	Acquisition des données du flux de transport de déchiffrement	9.6.2.3.5.4	S	C→H	Définit un filtre pour les acquisitions de section.
176	setDcrTsSectionAcqDefault	Acquisition des données du flux de transport de déchiffrement	9.6.2.3.5.3	S	C→H	Définit un filtre par défaut pour l'acquisition de section.
177	setEncrModes	Rechiffrement	9.7.2.5.2	S	C→H	Le **Micro serveur** informe l'**Hôte ECI** des modes (de chiffrement, de format de données et de synchronisation) qu'il prend en charge.
178	setPwrInfo	Gestion de la consommation d'énergie	9.4.7.2.3	S	C→H	Demande la notification des changements d'état de consommation d'énergie.
179	setUiClientAttention	Interface d'utilisateur	9.4.3.4.3	S	C→H	Le **Client ECI** demande à démarrer une session d'interface d'utilisateur sans association à un **Pointeur de média.**
180	setPwrWakeup	Gestion de la consommation d'énergie	9.4.7.3	S	C→H	Définit l'heure du réveil du **Client ECI**.

Appendice II

Domaines nécessitant des développements supplémentaires

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

Il a été établi que la présente Recommandation nécessite des développements et une validation supplémentaires afin de répondre aux exigences énoncées dans la Recommandation [UIT-T J.1010], et que la Recommandation [UIT-T J.1010] doit être mise à jour pour refléter les exigences de la spécification Enhanced Content Protection (ECP) de MovieLabs [b-ECP]. Les Recommandations [UIT-T J.1011], UIT-T J.1012, [UIT-T J.1013], [UIT-T J.1014], [UIT-T J.1015] et [b-ITU-T J.1015.1] devront à l'avenir être mises à jour pour refléter ces mises à jour de la Recommandation [UIT-T J.1010].

Plusieurs États Membres de l'UIT ainsi que des parties prenantes de divers secteurs – notamment des fabricants d'appareils et de composants électroniques, des propriétaires et des titulaires de licences de contenus protégés par le droit d'auteur, des fournisseurs de services over-the-top (OTT) et de télévision linéaire, et des fournisseurs de solutions de système d'accès conditionnel (CAS) et de gestion des droits numériques (DRM) – basées dans le monde entier se sont déclarés préoccupés par le fait que l'interface commune intégrée (ECI) ne répond pas pleinement aux exigences de la spécification ECP, ni aux exigences plus larges du secteur en matière de protection des contenus.

Plus précisément, leurs préoccupations ont été exprimées dans des contributions soumises à la réunion de la Commission d'étude 9 (CE 9) de l'UIT-T tenue du 16 au 23 avril 2020. Dans leurs contributions, Israël, l'Australie, Samsung (Membre de secteur de l'UIT-T) ainsi que Sky Group et MovieLabs (Associés de la CE 9) ont proposé d'apporter plusieurs modifications aux Recommandations relatives à l'interface ECI, mais aucun accord n'a été trouvé à leur sujet. Ces points sont répertoriés dans le document [b-CE 9 Rapport 17 Ann.1].

Les propositions visent à:

1) simplifier le système ECI en réduisant son champ d'application;
2) supprimer la gestion DRM;
3) supprimer le rechiffrement de contenu;
4) supprimer la gestion des logiciels;
5) ajouter des API pour les opérations de stockage et de chiffrement sécurisées;
6) autoriser des échelles de clés propres aux fournisseurs;
7) utiliser les exigences TEE J.1207;
8) inclure l'implémentation TEE pour les machines virtuelles;
9) augmenter la puissance des algorithmes de chiffrement, par exemple en utilisant SHA-384;
10) utiliser des certificats standard, comme UIT-T X.509;
11) revoir les communications entre clients;
12) mener des échanges supplémentaires avec l'ETSI;
13) effectuer une évaluation par les pairs supplémentaire;
14) envisager des alternatives au modèle de l'autorité de confiance;
15) définir plus précisément les aspects techniques des règles de conformité et de robustesse de l'interface ECI;
16) ajouter des exigences en matière de diversité, par exemple la randomisation de l'espace d'adresses;
17) ajouter des exigences relatives à la vérification de l'intégrité de l'exécution.

Ces propositions reflètent le fait que la protection des contenus et les menaces de compromission de contenu sont en constante évolution. L'interface ECI a été conçue initialement près de dix ans avant l'approbation de la présente Recommandation UIT-T. Des systèmes comme l'interface ECI doivent être évalués régulièrement en fonction à la fois des techniques d'attaque et des exigences de protection du secteur les plus récentes.

D'autres mécanismes existent pour assurer l'interopérabilité. En particulier, s'agissant de la gestion DRM, la plupart des services vidéo sur l'Internet ont déployé d'autres solutions pour assurer l'interopérabilité et répondre à leurs besoins.

Il est important d'apporter davantage de clarté, car de nombreux États Membres considèrent les normes de l'UIT comme des guides importants pour le développement de leurs marchés et de leurs secteurs. La liste de préoccupations vise à faire en sorte que la mise en œuvre de l'interface ICE sur les marchés nationaux puisse reposer sur une compréhension parfaite des conséquences de la présente Recommandation de l'UIT-T et que les questions soient prises en compte au moment d'examiner une législation, une réglementation ou des besoins du marché exigeant que les équipements de télévision numérique grand public soient interopérables. Elle vise également à faire en sorte que les fabricants d'équipements techniques, qui peuvent préférer utiliser un ensemble unique d'exigences ou d'autres normes pour concevoir les produits, puissent prendre en compte ces questions lors du développement de produits pour des marchés différents.

Bibliographie

[b-UIT-T J.1015.1] Recommandation UIT-T J.1015.1 (2020), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Système de sécurité évoluée – Bloc d'échelle de clés: authentification des informations sur les règles d'utilisation des mots de contrôle et des données associées 1*.

[b-UIT-T J Suppl. 7] Supplément 7 à la série J de Recommandations UIT-T (2020), *Interface commune intégrée pour les solutions CA/DRM interchangeables; Lignes directrices pour la mise en œuvre de l'interface commune intégrée*.

[b-CE 9 Rapport 17 Ann.1] Rapport de la réunion de la CE 9 de l'UIT-T, SG9-R17-Annexe 1 (2020), Annexe 1 au Rapport 17 de la réunion de la CE 9 organisée de manière entièrement virtuelle du 16 au 23 avril 2020. https://www.itu.int/md/T17-SG09-R-0017/en

[b-ETSI GS ECI 001-1] ETSI GS ECI 001-1, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 1: Architecture, Definitions and Overview*.

[b-ETSI GS ECI 001-2] ETSI GS ECI 001-2, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 2: Use cases and requirements*.

[b-ETSI GS ECI 001-3] ETSI GS ECI 001-3 V1.1.1 (2017-07), *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 3: CA/DRM Container, Loader, Interfaces, Revocation*.

[b-ETSI GS ECI 001-4] ETSI GS ECI 001-4, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 4: The Virtual Machine*.

[b-ETSI GS ECI 001-5-1] ETSI GS ECI 001-5-1, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions Part 5: The Advanced Security System Sub-part 1: ECI specific functionalities*.

[b-ETSI GS ECI 001-5-2] ETSI GS ECI 001-5-2, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 5: The Advanced Security System; Sub-part 2: Key Ladder Block.*

[b-ETSI TS 102 034] ETSI TS 102 034 (V1.4.1), *Digital Video Broadcasting (DVB); Transport of MPEG-2 TS Based DVB Services over IP Based Networks*.

[b-Richardson] Richardson, S. Ruby (2007), *RESTfull Web services*, L. o'Reilly.

[b-DASH-IF V3] Dash Industry Forum (2015), *Guidelines for Implementation: Dash-IF Interoperability Points version 3.0*.

[b-DASH-IF ID] Dash Industry Forum: "Identifiers for protection". http://dashif.org/identifiers/protection/.

[b-CA Browser] CA Browser Forum: "Baseline Requirements: Certificate Policy for the Issuance and Management of Publicly-Trusted Certificates". https://cabforum.org/.

[b-NIST SP 800-52r2] NIST SP 800-52 rev2 (August 2019), *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations*.

[b-CI Plus] CI Plus Specification V1.3.1 (2011-09).
Available at http://www.ci-plus.com.

[b-DLNA] DLNA Networked Device Interoperability Guidelines, Digital Living Network Alliance.
http://www.dlna.org/guidelines

[b-HbbTV] Hybrid Broadcast Broadband Television (HbbTV®) Operator Applications.

[b-ETSI GS ECI 001-6] ETSI GS ECI 001-6, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; Part 6: Trust Environment*.

[b-ETSI GS ECI 002] ETSI GS ECI 002, *Embedded Common Interface (ECI) for exchangeable CA/DRM solutions; System validation*.

[b-IETF RFC 8259] IETF RFC 8259 (2017), *The JavaScript Object Notation (JSON) Data Interchange Format*.

[b-IANA] IANA "Media Types" database.
http://www.iana.org/assignments/media-types/media-types.xhtml

[b-HDCP2.3] Digital Content Protection LLC (2018), *High Bandwidth Digital Content Protection System, Mapping HDCP to HDMI* revision 2.3., Feb 28
https://www.digital-cp.com/sites/default/files/HDCP%20on%20HDMI%20Specification%20Rev2_3.pdf

[b-Illgner] Klaus Illgner, Christoph Schaaf, Marnix Vlot (2016), *Embedded Common Interface (ECI) for Digital Broadcasting Applications: Security and Interoperability combined*, Broadband Journal of the SCTE, Vol. 38, No. 3, August.

[b-Menezes] Menezes, A., van Oorschot, P. and Vanstone, S (1996), *Handbook of Applied Cryptography*, CRC Press.

[b-ECP] MovieLabs Specification for Enhanced Content Protection – Version 1.2
https://movielabs.com/ngvideo/MovieLabs_ECP_Spec_v1.2.pdf

Les liens hypertextes du présent paragraphe étaient actifs au moment de la publication, mais leur validité à long terme ne peut être garantie.

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série D	Principes de tarification et de comptabilité et questions de politique générale et d'économie relatives aux télécommunications internationales/TIC
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	**Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias**
Série K	Protection contre les perturbations
Série L	Environnement et TIC, changement climatique, déchets d'équipements électriques et électroniques, efficacité énergétique; construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	Gestion des télécommunications y compris le RGT et maintenance des réseaux
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 et mesures et tests associés
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, communication entre systèmes ouverts et sécurité
Série Y	Infrastructure mondiale de l'information, protocole Internet, réseaux de prochaine génération, Internet des objets et villes intelligentes
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2020