Unión Internacional de Telecomunicaciones

UIT-T J.1012

(04/2020)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE J: REDES DE CABLE Y TRANSMISIÓN DE PROGRAMAS RADIOFÓNICOS Y TELEVISIVOS, Y DE OTRAS SEÑALES MULTIMEDIA

Acceso condicional y protección – Soluciones de acceso condicional insertadas e intercambiables y de gestión digital de los derechos

Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; contenedor, cargador, interfaces y revocación CA/DRM

Recomendación UIT-T J.1012

Unión Internacional de Telecomunicaciones

Recomendación UIT-T J.1012

Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; contenedor, cargador, interfaces y revocación CA/DRM

Resumen

La Recomendación UIT-T J.1012 forma parte de un conjunto de publicaciones que abarca el contenedor, el cargador, las interfaces del acceso condicional/gestión de derechos digitales (CA/DRM) y la revocación CA/DRM de la interfaz común integrada para soluciones CA/DRM intercambiables.

Esta Recomendación del UIT-T es una transposición de la norma ETSI GS ECI 001-3, y es el resultado de la colaboración entre la CE 9 del UIT-T y el ISG ECI del ETSI. Se han introducido modificaciones en las cláusulas 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, I-2 y en la Bibliografía. También se han efectuado algunas correcciones editoriales.

Historia

Edición	Recomendación	Aprobación	Comisión de Estudio	ID único*
1.0	UIT-T J.1012	23-04-2020	9	11.1002/1000/13573

Palabras clave

CA, DRM, intercambio.

* Para acceder a la Recomendación, sírvase digitar el URL http://handle.itu.int/ en el campo de dirección del navegador, seguido por el identificador único de la Recomendación. Por ejemplo, http://handle.itu.int/11.1002/1000/11830-en.

PREFACIO

La Unión Internacional de Telecomunicaciones (UIT) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones y de las tecnologías de la información y la comunicación. El Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

La observancia de esta Recomendación es voluntaria. Ahora bien, la Recomendación puede contener ciertas disposiciones obligatorias (para asegurar, por ejemplo, la aplicabilidad o la interoperabilidad), por lo que la observancia se consigue con el cumplimiento exacto y puntual de todas las disposiciones obligatorias. La obligatoriedad de un elemento preceptivo o requisito se expresa mediante las frases "tener que, haber de, hay que + infinitivo" o el verbo principal en tiempo futuro simple de mandato, en modo afirmativo o negativo. El hecho de que se utilice esta formulación no entraña que la observancia se imponga a ninguna de las partes.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB en la dirección http://www.itu.int/ITU-T/ipr/.

ÍNDICE

Página

Introducción

Esta Recomendación UIT-T[1] es una transposición de la norma [b-ETSI GS ECI 001-3] del ETSI y es el resultado de la colaboración entre la CE 9 del UIT-T y el ETSI ISG ECI. Se han introducido modificaciones en las cláusulas 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, I-2 y en la Bibliografía. También se han efectuado algunas correcciones editoriales.

El objetivo de esta Recomendación es facilitar la interoperabilidad y la competencia en los servicios de comunicaciones electrónicas y, en particular, en el mercado de los dispositivos de radiodifusión y audiovisuales. Sin embargo, existen otras tecnologías disponibles que también pueden resultar adecuadas y beneficiosas, en función de las circunstancias de los Estados Miembros.

La protección de servicios y contenidos mediante el acceso condicional (CA) y la gestión de derechos digitales (DRM) son elementos fundamentales para un rápido crecimiento de los servicios de difusión y banda ancha digitales. Ello incluye la distribución de contenidos en alta definición (HD) y ultradefinición (UHD) a diversos tipos de equipos en locales del cliente (**CPE**)[2] a fin de proteger los modelos comerciales de los propietarios de contenidos y de los proveedores de servicios, incluidos los radiodifusores y los operadores de televisión de pago. Mientras que los sistemas de CA se centran principalmente en la protección del contenido distribuido por redes unidireccionales, como ocurre normalmente en un entorno de radiodifusión, los sistemas DRM tienen su origen en entornos de red bidireccionales y permiten el acceso a contenidos a través de dispositivos de usuarios certificados y autenticados, normalmente con declaraciones de derechos completos sobre contenidos. En la práctica, no es posible hacer en todos los casos una clara distinción entre las funcionalidades CA y DRM y, por lo tanto, en la presente Recomendación se utiliza el término CA/DRM.

Las soluciones CA/DRM actualmente implantadas, ya sean integradas o en forma de hardware conectable, generan con frecuencia restricciones de uso a los proveedores de servicio/plataformas por un lado y a los consumidores por otro. Ello significa para los consumidores dependencias con respecto a la red, el servicio y los proveedores de contenidos a los que se aplican, y condiciona los **CPE** que resultan adecuados para los servicios de radiodifusión digital clásica, TV por el Protocolo Internet (TVIP) o servicios superpuestos (OTT). Aunque los **CPE** con funcionalidades CA o DRM integradas que son propiedad exclusiva de una plataforma vinculan un cliente al operador de esa plataforma en concreto, los módulos de hardware conectables permiten utilizar **CPE** disponibles en el mercado al por menor, como por ejemplo, cajas de medios integrados (STB, *set top boxes*) y conjuntos de TV integrados (iDTV). Debido a su formato y costo, los módulos de hardware conectables no cumplen requerimientos futuros, en particular los relativos al consumo de contenidos protegidos en tabletas y dispositivos móviles y en relación con los despliegues en los que el costo es un factor crítico.

Las soluciones tecnológicas actualmente utilizadas limitan la libertad de muchos actores de mercados de contenidos multimedios digitales. Gracias a los avances tecnológicos, existen en la actualidad soluciones software de CA/DRM innovadoras. Al maximizar la interoperabilidad manteniendo un alto nivel de seguridad, estas soluciones aplican un enfoque prometedor con vistas a futuras demandas del mercado, permiten el desarrollo de nuevas actividades comerciales y amplían las alternativas del consumidor para acceder a contenidos a través de la radiodifusión y las conexiones de banda ancha.

Los consumidores están interesados en seguir utilizando los **CPE** que han adquirido para su uso personal, por ejemplo, después de una mudanza o de un cambio de proveedor de servicios o, incluso, para obtener servicios de diferentes portales de vídeo comerciales. Para ello es necesaria la interoperabilidad entre los sistemas de CA y DRM de los **CPE** sobre la base de una arquitectura de seguridad adecuada. La única manera de impedir la fragmentación del mercado de los **CPE** y fomentar la competencia es garantizar que los sistemas de CA y DRM puedan intercambiarse de

---

[1] En el Apéndice II se indican varios aspectos susceptibles de mejora.

[2] El uso de la letra negrita en el texto de esta Recomendación indica términos con definiciones específicas para el contexto de la interfaz común integrada que pueden diferir del uso común.

manera sencilla y flexible en función del contexto, de una forma ligada a entornos de seguridad basados en las tecnologías más modernas.

Los operadores de plataformas están interesados en que la tecnología de seguridad pueda desplegarse de una forma adaptada al contexto y sea fácilmente gestionable a través de distintas redes y en todo tipo de dispositivos. La ventaja de una actualización sin solución de continuidad de los dispositivos existentes con los últimos sistemas de seguridad constituye una oportunidad comercial sin igual.

Un **Ecosistema ECI,** tal como se especifica en esta Recomendación y conforme al producto final basado en múltiples componentes de la **ECI**, contempla atributos importantes, como la flexibilidad y la escalabilidad, gracias a su implementación en software y la intercambiabilidad que garantiza soluciones válidas futuras y supone un impulso a la innovación. Otros aspectos adicionales son su aplicabilidad a contenidos distribuidos a través de varios tipos de redes, incluyendo los servicios de difusión clásicos, la TVIP o los OTT. Al impulsar el desarrollo del mercado, la especificación del sistema **ECI** para un ecosistema abierto ofrece la base para la intercambiabilidad de sistemas de CA y DRM en los **CPE** al menor costo posible para los consumidores y con restricciones mínimas para los vendedores de sistemas CA o DRM a la hora de desarrollar sus productos para el mercado de la televisión de pago.

Además de la Parte 4 de este conjunto de publicaciones, relativa a la máquina virtual, y la Parte 5 sobre la seguridad avanzada, la presente Recomendación, que constituye la Parte 3, especifica todos los elementos esenciales para la descarga e intercambio de clientes CA/DRM (**Clientes ECI**) y su entorno de ejecución (**Anfitrión ECI**) en un entorno de confianza que incluye la comunicación con las entidades funcionales necesarias mediante las API que se especifican detalladamente en el mismo.

Recomendación UIT-T J.1012

Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; contenedor, cargador, interfaces y revocación CA/DRM

1 Alcance

La arquitectura del **sistema ECI** se define en [UIT-T J.1011]; véase también [b-ETSI GS ECI 001-1]. El **sistema ECI** se basa en requisitos definidos en [UIT-T J.1010]; véase también [b-ETSI GS ECI 001-2]. En esta Recomendación se especifica la funcionalidad principal de un **Ecosistema ECI**, incluyendo información sobre el contenedor, el cargador, las interfaces y la revocación CA/DRM; véase también [b-Illgner]. Una característica innovadora y ventaja destacable del **Ecosistema ECI** en comparación con sistemas actualmente implantados es el hecho de disponer de una arquitectura completa basada en software para la carga de sistemas CA/CDR y el intercambio de los mismos, sin necesidad de módulos hardware conectables. Los contenedores software proporcionan un entorno seguro ("Sandbox") para los núcleos (*kernel*) de los sistemas CA o DRM, que a lo largo de este documento se denominan **Clientes ECI**, junto con sus instancias individuales de **máquina virtual**. Se especifican con detalle las interfaces de programación de aplicaciones (API) necesarias y pertinentes entre los **Clientes ECI** y el **Anfitrión ECI** que garantizan el funcionamiento de múltiples **Clientes ECI** en un entorno de operacional seguro y de forma completamente aislada del resto del firmware de los **CPE**. La instalación de un **Anfitrión ECI** y el intercambio que este lleva a cabo, así como de los múltiples **Clientes ECI**, es una labor del Cargador ECI, cuya carga inicial se realiza mediante un cargador de circuitos integrados. El **Anfitrión ECI** y los **Clientes ECI** se descargan a través del carrusel de datos de la radiodifusión digital de vídeo (DVB) para servicios de difusión y/o a través de mecanismos basados en IP desde un servidor en caso de accesos de banda ancha. Este proceso se integra en un entorno seguro y confiable, lo que ofrece una jerarquía de confianza para la instalación del **Anfitrión ECI** y los **Clientes ECI** y los intercambios entre ellos, que dispensa una protección eficaz contra ataques a la integridad y de sustitución. Por este motivo, el **Ecosistema ECI** integra un mecanismo de seguridad avanzado basado en el procesamiento avanzado y eficaz de palabras de control (CW), que se especifica como **Bloque de escalera de claves** y que se integra en un hardware del sistema sobre circuitos integrados (SoC) a fin de proporcionar la máxima seguridad necesaria para la conformidad con la **ECI**. Las funciones de Seguridad avanzada específicas de la **ECI** también juegan un papel fundamental en el proceso de reencriptación en caso de contenido protegido almacenado y/o asociado con la exportación de contenido protegido a un dispositivo externo que puede o no ser conforme con la **ECI**. Un sistema microDRM avanzado proporciona la funcionalidad necesaria y es parte integral de dicho concepto. La funcionalidad de Seguridad avanzada es pertinente asimismo en caso de revocación de un **CPE** o de un **Cliente ECI** específico. En la presente Recomendación se especifican las API conexas, tratándose la Seguridad avanzada con detalle en [UIT-T J.1014] y [UIT-T J.1015], véase asimismo [b-ETSI GS ECI 001-5-1] y [b-ETSI GS ECI 001-5-2].

El **Ecosistema ECI** está caracterizado por numerosas API, que garantizan la comunicación con las entidades pertinentes conexas, por ejemplo cargadores **ECI**, la importación y exportación de contenido protegido, la seguridad avanzada, la encriptación y desencriptación, las facilidades de almacenamiento local y la inserción de filigranas. También existen API adicionales para la interfaz persona-máquina (MMI) del **Cliente ECI** o para un lector opcional de **Tarjetas inteligentes**.

En caso de la necesidad de actualizaciones, el **Usuario** puede iniciar el intercambio de **Clientes ECI** o este puede ser solicitado por un operador. Se soportan un mínimo de dos **Clientes ECI**, pudiendo añadirse dos **Clientes ECI** adicionales en la medida en que exista capacidad de almacenamiento local en un grabador de video personal (PVR) o por motivos de exportación.

La presente Recomendación incluye especificaciones detalladas en las cláusulas que se indican a continuación.

En la cláusula 5 se especifica el sistema de certificados **ECI**, que incluye **Certificados** para varios propósitos como para el **Certificado de Cargador de Anfitrión ECI**, de **Cliente ECI** y de **Operador ECI**, así como la definición de dichos **Certificados**, y la **Lista de Revocación** asociada, su composición en forma de cadenas y la estructura del **certificado raíz**.

El **Cargador de Anfitrión ECI** se trata en la cláusula 6, donde el proceso de carga del **Anfitrión ECI** aborda el almacenamiento de una imagen, la verificación de su autenticidad por el **CPE** utilizando los datos de autenticación proporcionados por la Autoridad de confianza de la ECI (**TA ECI)** y la subsiguiente activación de la imagen. Se incluye la especificación del formato de fichero, el protocolo de transporte y la revocación por parte del **Operador** de las **imágenes del Anfitrión ECI**.

La cláusula 7 incluye la especificación completa del **Cargador de Cliente ECI**, basada en la capacidad del **Anfitrión ECI** de descargar, almacenar y activar **Imágenes de Cliente ECI** y datos conexos. El proceso de carga de un **Cliente ECI** puede dividirse en varias fases, desde el descubrimiento a la descarga e inicialización de los **Clientes ECI**, pudiendo realizarse la descarga de datos del flujo de difusión o de Internet.

La cláusula 8 aborda la especificación de la revocación, incluyendo la funcionalidad necesaria para excluir selectivamente la prestación de servicios a los **CPE** basada en el estado de la **TA ECI** del hardware del **CPE**, el **Anfitrión ECI**, otras **Operaciones de plataforma** y **Clientes ECI** descargados.

En la cláusula 9 se especifican detalladamente las interfaces del **Cliente ECI**, que entre otras cosas, incluyen una especificación muy completa del ecosistema **ECI**, las API relativas a los recursos generales del **Anfitrión ECI**, los recursos propios de la ECI del **Anfitrión ECI**, los recursos de desencriptación del **Anfitrión ECI**, los recursos de reencriptación del **Anfitrión ECI**, los recursos relacionados con la protección del contenido y los recursos relativos a la relación entre **Clientes ECI**.

Finalmente, la cláusula 10 aborda las funcionalidades obligatorias y facultativas del **Anfitrión ECI**.

La presente especificación de los elementos fundamentales de la **ECI** sólo se aplica a la recepción y ulterior procesamiento del contenido, controlado por un sistema de acceso condicional y/o un sistema de gestión de derechos digitales y que ha sido encriptado por el proveedor de servicios.

Queda fuera del alcance de la presente Recomendación los contenidos no controlados por un sistema de acceso condicional y/o DRM.

La presente Recomendación debe utilizarse conjuntamente con un marco contractual (acuerdo de licencia), normas de cumplimiento y robustez y acuerdos del proceso de certificación adecuados bajo el control de una autoridad de confianza, que no son objeto de especificaciones técnicas como las del Grupo de especificaciones de la **ECI**. Alguno de estos aspectos básicos puede encontrarse en un anexo informativo a [b-ETSI GS ECI 001-6], relativa al entorno de confianza, que especifica los mecanismos técnicos y las relaciones en el marco de un entorno de confianza.

2 Referencias

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes. En esta Recomendación, la referencia a un documento, en tanto que autónomo, no le otorga el rango de una Recomendación.

[UIT-T J.1010] Recomendación UIT-T J.1010 (2016), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; Casos y requisitos de utilización*.

[UIT-T J.1011] Recomendación UIT-T J.1011 (2016), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; Arquitectura, definiciones y visión general*.

[UIT-T J.1013] Recomendación UIT-T J.1013 (2020), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; máquina virtual*.

[UIT-T J.1014] Recomendación UIT-T J.1014 (2020), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; seguridad avanzada – funcionalidades ECI*.

[UIT-T J.1015] Recomendación UIT-T J.1015 (2020), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; sistema de seguridad avanzado – Bloque de escalera de claves*.

[UIT-T T.871] Recomendación UIT-T T.871 (2011), *Tecnología de la información – Compresión digital y codificación de imágenes fijas de tonos continuos: formato de intercambio de ficheros JPEG (JFIF).*

[ISO/CEI 23001-7] ISO/CEI 23001-7:2015, *Information technology – MPEG systems technologies – Part 7: Common encryption in ISO base media file format files*.

[ISO/CEI 23009-1] ISO/CEI 23009-1:2014, *Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*.

[ISO/CEI 13818-1-1] ISO/CEI 13818-1-1:2007, *Information technology – Generic coding of moving pictures and associated audio information – Part 1: Systems*.

[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, *Information technology – Document Container File – Part 1: Core.*

[IETF RFC 4122] IETF RFC 4122 (julio de 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 System Software Update 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, *Information technology – Generic coding of moving pictures and associated audio information – Part 6: Extensions for 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, *Identification cards – Integrated circuit cards – Part 1: Cards with contacts – Physical Characteristics*.

[ISO/CEI 7816-2] ISO/CEI 7816-2, *Identification cards – Integrated circuit cards – Part 2: Cards with contacts – Dimensions and location of the contacts*.

[ISO/CEI 7816-3] ISO/CEI 7816-3, *Identification cards – Integrated circuit cards – Part 3: Cards with contacts – Electrical Interface and transmission protocols*.

[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, *Identification cards – Integrated circuit cards – Part 3: Cards with contacts – Registration of application providers*.

[ISO/CEI 7810] ISO/CEI 7810, *Identification cards – Physical characteristics*.

[ISO/CEI 23001-9] ISO/CEI 23001-9:2014, *Information Systems – MPEG system technologies – Part 9: Common Encryption of MPEG2 transport streams*.

[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, *Information technology – Coding of audio-visual objects – Part 12: ISO base media format*.

[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] W3C Recommendation (2003), *Portable Network Graphics (PNG) Specification (Segunda edición)*.

[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, *Information technology – 8-bit single-byte coded graphic character sets, Part 1: Latin alphabet No. 1*.

[ISO 3166-1] ISO 3166-1:2006, *Codes for the representation of names of countries and their subdivisions – Part 1: Country codes*.

[ISO 639-2] ISO 639-2:1998, *Codes for the representation of names of languages – Part 2: Alpha-3 code*.

[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, *Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange*.

3 Definiciones

3.1 Términos definidos en otros documentos

Ninguno.

3.2 Términos definidos en esta Recomendación

En esta Recomendación se definen los siguientes términos:

La utilización en esta Recomendación de términos que figuran en letra negrita y que comienzan por letra mayúscula refleja que se trata de términos que tienen un significado específico para la ECI que puede diferir del uso común de los mismos.

3.2.1 sistemas de seguridad avanzada (sistema AS): Función de un CPE conforme con **ECI**, que ofrece funciones de seguridad mejoradas (hardware y software) para un **Cliente ECI**.

3.2.2 intervalo de seguridad avanzada (AS): Recursos del bloque de seguridad avanzada que un **Anfitrión ECI** proporciona exclusivamente a un **Cliente ECI**.

3.2.3 sesión de intervalo de seguridad avanzada: Recursos y cálculo en un intervalo de **AS** relacionados con la desencriptación o reencriptación de un elemento de contenido.

3.2.4 hermano: Otro **Hijo** del mismo **Padre**.

NOTA – **Padre**, **Hijos**, **Hermano**, hacen referencia a entidades que gestionan **Certificados**.

3.2.5 certificado: Estructura de datos que se define en la cláusula 5 de esta Recomendación con una firma digital segura complementaria que identifica una **Entidad**.

NOTA – El titular de la clave secreta de la firma atestigua la corrección de los datos (los autentica) firmándolos con su clave secreta. Su clave pública puede utilizarse para verificar los datos.

3.2.6 cadena de certificados: Lista de **Certificados** que se autentican unos a otros y que incluye una lista de revocación raíz.

3.2.7 subsistema de procesamiento de certificado (CPS): Subsistema del **Anfitrión ECI** que realiza el procesamiento de la verificación del **Certificado** y que proporciona una robustez adicional contra la manipulación indebida.

3.2.8 hijo, hijos: **Entidad** (entidades) a las que hace referencia un **Certificado** firmado por un **Padre** (común).

NOTA – **Padre, Hijos, Hermano** hacen referencia a entidades que gestionan **Certificados**: datos y software de inicialización utilizados para arrancar el SoC de un CPE.

3.2.9 sistema de protección del contenido: Sistema de un **Ecosistema ECI** que emplea técnicas criptográficas para gestionar el acceso al contenido y los servicios.

NOTA – El término puede intercambiarse frecuentemente con el sistema alternativo de protección del servicio. Son sistemas típicos de este tipo los sistemas de acceso condicional (CAS) o los sistemas de gestión de derechos digitales (DRM).

3.2.10 equipo en los locales del cliente (CPE): Receptor de medios que implementa la **ECI**, y que permite al **Usuario** el acceso a servicios de medios digitales.

3.2.11 fabricante de CPE: Empresa que fabrica **CPE** conformes con la **ECI**.

3.2.12 interfaz común integrada (ECI): Arquitectura y sistema especificado por el ISG de ETSI "Embedded CI" (CI integrada), que permite la creación e instalación en equipos de cliente (**CPE**) de **Clientes ECI** intercambiables basados en software y que, por tanto, permiten la interoperabilidad de dispositivos **CPE** con relación a la **ECI**.

3.2.13 aplicación ECI: Aplicación basada en HTML residente en un **Cliente ECI**, y que se ejecuta en una sesión del navegador destinada a interactuar con el **Usuario** y proporcionar información del **Usuario** al **Cliente ECI**.

3.2.14 fabricante de circuitos integrados ECI: Empresa que proporciona sistemas en un conjunto de circuitos integrados que implementa la funcionalidad especificada para la **ECI**.

3.2.15 cliente de la interfaz común integrada (Cliente ECI): Implementación de un cliente CA/DRM conforme a las especificaciones de la interfaz común integrada.

NOTA – El módulo software del **CPE** proporciona los medios para recibir de manera protegida y para controlar la ejecución de los derechos y prerrogativas del consumidor relativos al contenido distribuido por un distribuidor de contenidos u **operador**. También recibe las condiciones bajo las cuales el consumidor puede utilizar un derecho o prerrogativa, además de las claves para desencriptar los diversos mensajes y contenidos.

3.2.16 imagen de cliente ECI: Fichero software de código de máquina virtual (VM) y datos de inicialización requeridos por el **Cargador de Cliente ECI**.

3.2.17 cargador de cliente ECI: Módulo software que forma parte del **Anfitrión ECI** que permite la descarga, verificación e instalación de un nuevo software de **Cliente ECI** en un **Contenedor ECI** del **Anfitrión ECI**.

3.2.18 contenedor ECI: Instancia de máquina virtual (VM) individual que tiene bibliotecas de apoyo complementarias y una API **ECI** que permite que una única instancia de un **Cliente ECI** se ejecute en un **CPE**.

3.2.19 ecosistema ECI: Operativa comercial que consta sobre el terreno de una **TA** y varias plataformas y **CPE** conformes con la **ECI**.

3.2.20 anfitrión ECI: Sistema hardware y software de un **CPE**, que abarca las funcionalidades relativas a la **ECI** y que tiene interfaces con el **Cliente ECI**.

NOTA – El **Anfitrión ECI** forma parte del firmware del **CPE**.

3.2.21 imagen de anfitrión ECI: Fichero o ficheros con software y datos de inicialización para un entorno **ECI**.

NOTA 1 – Una imagen de **Anfitrión ECI** puede constar de un conjunto de ficheros **Imagen de Anfitrión ECI**.

NOTA 2 – También puede contener otro software que no cause interferencia al Anfitrión ECI o permita la observación no deseada del mismo.

3.2.22 cargador de anfitrión ECI: Módulo software, que permite la descarga, verificación e instalación del software de **Anfitrión ECI** en un **CPE**.

NOTA – En una configuración de descarga en varias fases este término se utiliza para hacer referencia a todas las funciones de descarga críticas desde el punto de vista de la seguridad que participan en la carga del **Anfitrión ECI**.

3.2.23 certificado raíz ECI: **Certificado** emitido para verificar elementos aprobados por una **TA ECI**.

3.2.24 entidad: Organización (por ejemplo, fabricante, **operador** o **vendedor de seguridad**) o elemento del mundo real (por ejemplo, **Anfitrión ECI**, **Operación de Plataforma** o **Cliente ECI**) identificado mediante un ID único en un **Ecosistema ECI**.

3.2.25 cadena de exportación: Cadena de certificados utilizada para la exportación a uno o a un grupo de **sistemas microDRM**.

3.2.26 conexión de exportación: Relación autenticada entre un **Cliente ECI** que puede desencriptar contenido y un **Microservidor** que puede reencriptar contenido.

3.2.27 grupo de exportación: Grupo de **sistemas microDRM** a los que se permite la exportación.

3.2.28 padre: Firmante del **Certificado** de la **Entidad Hijo**.

NOTA – **Padre, Hijos, Hermano** hacen referencia a entidades que gestionan **Certificados**.

3.2.29 series de imágenes: Series de imágenes para un **Anfitrión ECI** o un **Cliente ECI** que son diferentes en función del identificador (**CPE**_id) del **CPE**, y sin embargo representan (casi) la misma funcionalidad.

3.2.30 cadena de importación: Cadena desde el POPK de un **Cliente ECI** hasta una **Entidad** que representa un sistema de exportación o un **Grupo de exportación**.

NOTA – Una **Cadena de exportación** y su correspondiente **Cadena de importación** pueden utilizarse para autenticar una sesión de **Microservidor** en la que se importa contenido para un **Cliente ECI** de exportación.

3.2.31 conexión de importación: Conexión aprobada desde un **Cliente ECI** a un **Microservidor** que le permite importar contenido desencriptado para la subsiguiente reencriptación del mismo.

3.2.32 fabricante: Entidad que diseña y vende los CPE, que incorporan una implementación del sistema **ECI** y permiten instalar **Anfitriones ECI** y **Clientes ECI** mediante la descarga de software.

3.2.33 distintivo de medios: Referencia a una configuración para el procesamiento de un programa de desencriptación o reencriptación entre un **Cliente ECI** y un **Anfitrión ECI**.

3.2.34 microcliente: **Cliente ECI** o no **ECI** que puede desencriptar contenido que ha sido reencriptado por un **Microservidor**.

3.2.35 microservidor: **Cliente ECI** que puede importar contenido desencriptado, reencriptarlo de nuevo y autenticar un **Cliente ECI** específico o un grupo de **Clientes ECI** como **Objetivo** de una ulterior desencriptación.

3.2.36 sistema microDRM: **Sistema de protección de contenido** que reencripta contenido en un **CPE** con un microservidor y que permite la decodificación de ese contenido reencriptado por **Microclientes** autenticados.

NOTA – El **Microservidor** y los **Microclientes** son provisionados por un operador de **Sistema microDRM**.

3.2.37 operador: Organización que proporciona **Operaciones de la plataforma** que la **TA ECI** incluye como autorizados para la firma del **Ecosistema ECI**.

NOTA – Un **Operador** puede realizar múltiples **Operaciones de plataforma**.

3.2.38 operación de plataforma (PO): Instancia específica de una operación de prestación de un servicio técnico con una única identidad **ECI** a los efectos de seguridad.

3.2.39 sesión de reencriptación: Proceso controlado por un **Microservidor** para la importación de contenido de una **Conexión de importación**, su reencriptación y la producción de la información de desencriptación que el **Objetivo** autenticado necesita para su ulterior desencriptación.

3.2.40 petición: Mensaje desde un emisor a un receptor por el que solicita determinada información o para realizar una determinada operación en un **Ecosistema ECI**, que se especifica en los campos de datos de la petición.

NOTA – Para más información véase la cláusula 9.2.3.

3.2.41 contestación: Mensaje en un **Ecosistema ECI** que responde a una **petición**.

NOTA – Para más información véase la cláusula 9.2.3.

3.2.42 lista de revocación (RL): Lista de **Certificados** que han sido revocados y que por lo tanto no deben seguir utilizándose.

3.2.43 raíz: Clave pública o **Certificado** que contiene una clave pública que sirve de base para la autenticación de una cadena de **Certificados**.

3.2.44 canal autenticado seguro (SAC): Trayecto de comunicación (canal) establecido entre dos **Entidades** identificadas entre ellas de forma segura (autenticado) y que han acordado la encriptación de los datos transferidos entre ellas (seguro).

3.2.45 servicio: Contenido proporcionado por una **Operación de Plataforma**.

NOTA – En el contexto **ECI** solo se considera contenido protegido.

3.2.46 clave pública de emisor (SPK): Clave pública del emisor del contenido encriptado utilizada en un **Ecosistema ECI** para verificar el origen de la firma de la primera clave de una cadena de claves utilizadas para desencriptar el contenido, siendo el emisor parte de una **Operación de Plataforma**.

3.2.47 tarjeta inteligente: Dispositivo de seguridad hardware desconectable utilizado por diversos proveedores de CA o DRM para mejorar el nivel de seguridad de sus productos en un **Ecosistema ECI**.

3.2.48 objetivo: **Microcliente** o grupo de **Microclientes** cuyo contenido reencripta un **Microservidor**.

3.2.49 autoridad de confianza (TA): Organización que rige en materia de normas y reglamentaciones aplicables a determinadas implementaciones de **ECI** y que tiene por objetivo un mercado concreto.

NOTA – La **Autoridad de confianza** debe ser una entidad jurídicamente establecida que pueda hacer reclamaciones de carácter jurídico. La **Autoridad de confianza** debe ser imparcial para con todos los agentes del **Ecosistema ECI** que gobierna.

3.2.50 tercera parte de confianza (TTP): Proveedor de servicios de seguridad que emite **Certificados** y claves para **Fabricantes** conformes de los componentes pertinentes de un **Sistema ECI**.

NOTA – Está bajo el control de la **Autoridad de confianza (TA)**.

3.2.51 usuario: Persona que opera un dispositivo conforme con **ECI**.

3.2.52 instancia de máquina virtual (VM): Instanciación de una VM establecida por un **Anfitrión ECI** y que para un **Cliente ECI** es un entorno de ejecución de la misma.

4 Siglas y acrónimos

En esta Recomendación se utilizan las siguientes siglas y acrónimos:

4CC	Código de cuatro caracteres (también se usa *FourCC*)
3DES	Triple DES
AEAD	Encriptación autenticada y datos asociados (*authenticated encryption with associated data*)
AES	Norma de encriptación avanzada (*advanced encryption standard*)
AES-GCM	Modo contador de Galois de AES (AES *Galois counter mode*)
AID	IDentificador de aplicación (*application IDentifier*)
AK	Clave de autenticación (*authentification key*)
APDU	Unidad de datos del protocolo de aplicación (*application protocol data unit*)
API	Interfaz de programación de aplicaciones (*application programming interface*)
AS	Seguridad avanzada (*advanced security*)
ASCII	American Standard Code for Information Interchange
ATR	Respuesta a reinicialización (*answer to reset*)
BAT	Tabla de asociación de un paquete de programas (*bouquet association table*)
BMFF	Formato de fichero de medios de referencia (*base media file format*)
BSD	Distribución de software Berkeley (*Berkeley software distribution*)
CA	Acceso condicional (*conditional access*)
CA/DRM	Acceso condicional/gestión de derechos digitales (*conditional access/digital rights management*)
CAT	Tabla de acceso condicional (*conditional access table*)
CBC	Encadenamiento de bloques de cifrado (*cipher block chaining*)
CENC	Encriptación común (*common encryption*)
CI	Interfaz común (*common interface*)
CP	Propiedad del contenido (*content property*)

CPE	Equipo en los locales del cliente (*customer premises equipment*)
CPS	Subsistema de procesamiento del certificado (*certificate processing subsystem*)
CPU	Unidad central de proceso (*central processing unit*)
CRC	Verificación por redundancia cíclica (*cyclic redundancy check*)
CRL	Lista de Revocación de certificado (*certificate revocation list*)
CSA	Algoritmo de aleatorización común (*common scrambling algorithm*)
CSA1	Algoritmo de aleatorización común, primera versión
CSA3	Algoritmo de aleatorización común, tercera versión
CSS	Hojas de estilo en cascada W3C (W3C *cascading style sheets*)
CSS3	CSS versión 3
CTR	Modo contador (*counter mode*)
CW	Palabra de control (*control word*)
Dash	Flujo adaptable dinámico sobre HTTP (*dynamic adaptive streaming over HTTP*)
DDB	Bloque de datos de descarga (*download data block*)
DDOS	Denegación de servicio distribuida (*distributed denial of service*)
DES	Norma de encriptación de datos (*data encryption standard*)
DHE	Efemérides Diffie-Hellman (*ephemeral Diffie-Hellman*)
DII	Indicación de información de descarga (*download info indication*)
DLNA	Alianza de red para la vida digital (*Digital Living Network Alliance*)
DNS	Sistema de nombres de dominio (*domain name system*)
DRM	Gestión de derechos digitales (*digital rights management*)
DSI	Inicio del servidor de descarga (*download server initiate*)
DSMCC	Instrucciones y control de medios de almacenamiento digital (*digital storage media command and control*)
DVB	Difusión de video digital (*digital video broadcasting*)
EAC	Certificado de autorización de exportación (*export authorization certificate*)
EAOC	Certificado de operador de autorización de exportación (*export authorization operator certificate*)
ECM	Mensaje de control de prerrogativas (*entitlement control message*)
EGC	Certificado de grupo de exportación (*export group certificate*)
EIT	Tabla de información de eventos (*event information table*)
EMM	Mensaje de gestión de prerrogativas (*entitlement management message*)
ES	Flujo elemental (*elementary stream*)
ESC	Certificado de sistema de exportación (*export system certificate*)
GCM	Modo contador/Galois (*Galois/counter mode*)
GMT	Tiempo medio de Greenwich (*Greenwich mean time*)
HD	Alta definición (*high definition*)

HDCP	Protección de contenido digital de gran anchura de banda (*high-bandwidth digital content protection*)
HTML	Lenguaje de marcaje de hipertexto (*hyper text mark-up language*)
HTTP	Protocolo de transferencia de hipertexto (*hypertext transfer protocol*)
HTTP(S)	Protocolo seguro de transferencia de hipertexto (*hypertext transfer protocol secure*)
iDTV	Receptor de TV digital integrado (*integrated digital TV receiver*)
IFSC	Tamaño del campo de información de la tarjeta (*information field size of card*)
IFSD	Tamaño del campo de información del dispositivo (*information field size of device*)
IP	Protocolo Internet (*Internet protocol*)
IPTV	TV mediante el protocolo Internet (*TV using the Internet protocol (IP*))
IPv4	Protocolo Internet, versión 4
IPv6	Protocolo Internet, versión 6
ISO	Organización Internacional de Normalización (*International Organization for Standardisation*)
ISOBMFF	Formato de fichero de medios básico de ISO (ISO *base media file format*)
LAN	Red de área local (*local area network*)
LSB	Bit menos significativo (*least significant bit*)
Memoria NV	Memoria no volátil
MIME	Extensiones de correo internet mutipropósito (*multipurpose internet mail extensions*)
MMI	Interfaz persona máquina (*man machine interface*)
MP4	Formato de contenedor multimedios digital (*digital multimedia container format (también denominado MPEG-4 parte 14*)
MPD	Descripción de la presentación de medios (*media presentation description*)
MPEG	Grupo de Expertos en imágenes en movimiento (*motion picture experts group*)
MSB	Bit más significativo (*most significant bit*)
N.A.	No aplicable
NV	No volátil
OS	Sistema operativo (*operating system*)
OTT	Sistema superpuesto (sobre la Internet abierta) (*over the top (over the open Internet*))
OUI	Identificador único de organización (*organizationnally unique identifier*)
PAT	Tabla de asociación de programas (*program association table*)
PayTV	Televisión de pago (*pay television*)
PES	Flujo elemental de paquetes (*packet elementary stream*)
PID	Identificador de paquetes MPEG (MPEG *packet identifier*)
PIN	Número de identificación personal (*personal identification number*)
PKIX	Infraestructura de clave pública X.509 (*public-key infrastructure X.509*)
PMT	Tabla de mapa de programas (*program map table*)
PO	Operación de plataforma (*platform operation*)

POC	Certificado de operación de plataforma (*platform operation certificate*)
POPK	Clave pública de operación de plataforma (*platform operation public key*)
PPS	Selección de protocolo y parámetro (*protocol and parameter selection*)
PSI	Información específica de programa (*program specific information*)
PSSH	Encabezamiento específico del sistema de protección (*protection system specific header*)
PVR	Grabador de video personal (*personal video recorder*)
RAM	Memoria de acceso aleatorio (*random access memory*)
RFU	Reservado para uso futuro (*reserved for future use*)
RL	Lista de Revocación (*revocation list*)
SAC	Canal autenticado seguro (*secure authenticated channel*)
SDT	Tabla de descripción de servicio (*service description table*)
SHA	Algoritmo hash seguro (*secure hash algorithm*)
SI	Información de servicio (*service information*)
SIM	Modulo de identidad de abonado (*subscriber identity module*)
SoC	Sistema sobre circuitos integrados (*system on chip*)
SPK	Clave pública de firma (*signature public key*) (también denominada clave de verificación de firma (*signature verification key*))
SSK	Clave secreta de firma (*signature secret key*) (también denominada clave privada de firma (*signature private key*))
SSL	Capa de conector segura (*secure sockets layer*)
SSU	Actualización del software del sistema (*system software update*)
STB	Caja multimedios (*set top box*)
TA	Autoridad de confianza (*trust authority*)
TCK	Byte de verificación (*check byte*)
TCP	Protocolo de control de transmisión (*transmission control protocol*)
TLS	Seguridad en la capa de transporte (*transport layer security*)
TPDU	Unidad de datos del protocolo de transporte (*transport protocol data unit*)
TPEGC	Certificado de grupo de exportación de tercero (*third party export group certificate*)
TS	Flujo de transporte (*transport stream*)
TTP	Tercero confiable (*trusted third party*)
TV	TeleVisión
UDP	Protocolo de datagramas de usuario (*user datagram protocol*)
UHD	Definición ultra alta (*ultra high definition*)
UI	Interfaz de usuario (*user interface*)
uimsbf	Entero sin signo, con el bit más significativo en primer lugar (*unsigned integer, most significant bit first*)
UNT	Tabla de notificación de actualizaciones (*update notification table*)

URI	Información de derechos de uso (*usage rights information*)
URL	Localizador uniforme de recursos (*uniform resource locator*)
USB	Bus serie universal (*universal serial bus*)
UTF	Formato de transformación UCS (conjunto de caracteres universales) (*ucs (universal character set) transformation format*)
UUID	Identificador único universal (*universally unique identifier*)
VM	Máquina virtual (*virtual machine*)
WAN	Red de área amplia (*wide area network*)
WEB	World Wide Web

5 Sistema de Certificados ECI

5.1 Introducción

5.1.1 Alcance

La **ECI** utiliza **Certificados** para varios fines, tales como **Cargador de Anfitrión ECI**, **Cargador de Cliente ECI** y **Certificados de Operador ECI**. En esta cláusula se definen estos **Certificados** y la **Lista de Revocación** asociada, su composición en cadenas y la estructura del **Certificado Raíz**. La definición utiliza un formato binario compacto que se especifica en la presente Recomendación adaptada a la implementación hardware y a una criptografía adecuada, así como un sistema de señalización sencillo para versiones y extensiones futuras.

5.1.2 Notación y convenios relativos a los campos

Las definiciones de las estructuras de datos que se exponen a continuación se estructuran directamente en una secuencia de bytes. Cualquier función criptográfica se define para su funcionamiento conforme a la representación de la secuencia de bytes.

La definición de datos responde a un alineamiento natural de campos de 16 bytes y 32 bytes para simplificar el procesamiento de los datos en las CPU de 32 bits. El relleno de bits (*padding*) se utiliza como un campo genérico para indicar los campos que es necesario rellenar para este fin. Para ello se utiliza la función padding (n_bytes) siendo n_bytes el límite de la alineación expresado en número de bytes desde el inicio de la estructura de datos definida. Los campos de relleno se obviarán cuando se interpreten las estructuras de datos. El valor del campo de relleno se pondrá a cero (0).

Cualquier campo que se defina mediante otra estructura de datos mediante una definición de tipo no tiene nemónico. En general, para ese tipo de campo no se define la longitud del campo.

5.1.3 Campo extensión

Muchas de las estructuras de datos más importantes tienen un campo extensión que permite añadir futuras extensiones (retrocompatibilidad). La definición figura en el Cuadro 5.1.3-1.

Cuadro 5.1.3-1 – Definición del campo extensión

Sintaxis	N.° de bits	Mnemónico
`Extension_Field {`		
`  padding(4)`		
`  length`	32	uimsbf
`  for (i=0; i<length; i++) {`		
`  extension_byte`	8	uimsbf
`  }`		
`}`		

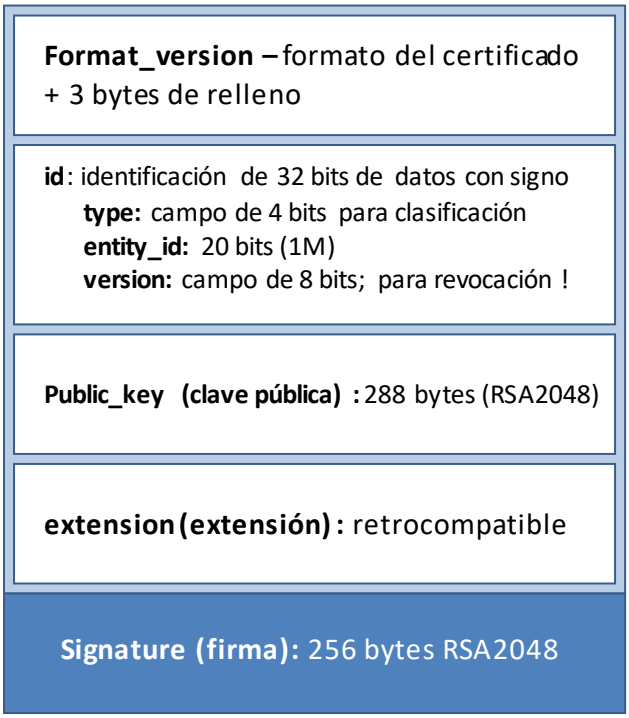
Semántica:

length: entero	Número de bytes del bucle siguiente. El valor debe ser múltiplo de 4, pero puede ser 0.
extension_byte: byte	Campo de datos con información que pueden ignorar las implementaciones basadas en versiones del presente documento que no hayan definido el contenido de este campo.

5.2 Certificados ECI

Un **Certificado ECI** tiene una estructura sencilla. A diferencia de los certificados X.509 utilizados en Internet, el ID del **Certificado** es simplemente un número binario destinado exclusivamente a ser interpretado por una máquina.

En la Figura 5.2-1 se muestra la estructura genérica de un **Certificado**.



J.1012(20)_F5.2-1

Figura 5.2-1 – Versión 1 del formato de Certificado ECI

El formato del **Certificado ECI** se define en el Cuadro 5.2-1.

Cualquier elemento con signo utilizará un campo de inicio diferenciado de 8 bytes, cuyo primer byte es el formato de la versión del elemento con signo, seguido (para los elementos de la versión 1) de 3 bytes de relleno, y 4 bytes que representan un ID único en el contexto de la clave secreta de la entidad firmante.

Cuadro 5.2-1 – Definición de un certificado ECI

Sintaxis	N.° de bits	Mnemónico
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 {		
format_version	8	uimsbf
if (version == 0x01) {		
ECI_Certificate_Data_v1 **data**		
ECI_Signature_v1 **signature**		
}		
}		

Semántica:

format_version: entero	Los valores 0x00, 0x02..0xFF están reservados. Valor 0x01: versión 1 del formato de **Certificado ECI**. Las implementaciones que no reconozcan un tipo de **Certificado** no lo procesarán y su respuesta a las peticiones de validación será que se ha producido un fallo.
id: entero	Identificación del Certificado en forma de número de 32 bits que es único en el contexto del **Padre Certificado** (firmante del **Certificado**). Los valores 0x00000 y 0xF0000-0xFFFFF están reservados.
type: entero	**type** define el tipo de Entidad, tal como **Fabricante, Anfitrión ECI, Operador**, etc. en el contexto del firmante (**Padre**). Los **Certificados** con un valor de tipo de 0x0…0x7 requerirán una **Lista de Revocación** para la verificación de los **Hijos**. Los valores de tipo 0x8 y superiores no requerirán una **Lista de Revocación** para la verificación de los **Hijos** (véase el Cuadro 5.2-2).
entity_id: entero	Define el número de la **Entidad**. El **entity_id** incluye varios subformatos según se define para cada tipo de **Certificado**. Salvo que se defina otra cosa, los id de entidad (id_entity) son únicos en el contexto del **Padre**/(firmante del **Certificado** o la **Lista de Revocación**).
version	Número de versión del **Certificado** de entidades, se asignan en orden ascendente (normalmente en incrementos de 1).
extension: Extension_Field	Los datos de este campo serán ignorados por las funciones de procesamiento que no hayan sido definidas para interpretarlo. Este campo puede utilizarse para datos específicos en aplicaciones concretas de la definición de **Certificado** genérica. Su interpretación es función del contexto. Las aplicaciones no conformes con la **ECI** no utilizarán este campo salvo que se especifique explícitamente que está permitido.
public_key: ECI_Public_Key_v1	Clave pública (asignada por el **Padre**) de la Entidad de este **Certificado**.
data: ECI_Certificate_Data	Esta es la sección de datos del **Certificado**.
signature: byte[256]	El campo firma contiene la representación de la secuencia de bytes de la firma del **Padre** del **Certificado**, utilizando las funciones criptográficas definidas en el Anexo A.

Cualquier verificación de un **Certificado ECI** incluirá la verificación de la longitud total del **Certificado** en función de acumulación de las definiciones de campos.

La mayoría de **Certificados** y **Listas de Revocación** utilizan valores genéricos a fin de garantizar la unicidad de los valores asignados. En el Cuadro 5.2-2 se ofrece una visión general de todos los datos con signo de la **TA ECI** (**Autoridad de confianza ECI**).

Cuadro 5.2-2 – Asignación de ID y Padres para elementos con signo

Padre	Tipo	Campo ID	Descripción
Raíz	0x0	0xFFFFF	Raíz
Raíz	0x1	Id de Fabricante, <> 0xFxxxx	**Certificado** de Fabricante
Raíz	0x1	Id de RL de Fabricante == 0xFxxxx	**Lista de Revocación** de Fabricante
Fabricante	0x0	Id de Anfitrión, <> 0xFxxxx	**Certificado de Anfitrión ECI**
Fabricante	0x0	RL de Anfitrión, == 0xFxxxx	**Lista de Revocación de Anfitrión ECI**
Anfitrión	0x8	Id de imagen de Anfitrión	**Imagen de Anfitrión ECI**
Anfitrión	0x9	Id de series de imágenes de Anfitrión	**Certificado de Series de Imágenes de Anfitrión ECI**
Series de imágenes de anfitrión	0x9	Id de Objetivo de imagen	Imágenes de series de **Anfitriones ECI**
Raíz	0x2	Id de Suministrador, <> 0xFxxxx	**Certificado del suministrador de seguridad**
Raíz	0x2	Id de RL de Suministrador == 0xFxxxx	**Lista de Revocación del suministrador de seguridad**
Suministrador	0x0	Id de Cliente <> 0xFxxxx	**Certificado de Cliente ECI**
Suministrador	0x0	RL de Cliente, == 0xFxxxx	**Cliente ECI** y **Lista de Revocación** de series de **Clientes ECI**
Cliente	0x0	Id de Cliente	**Imagen de Cliente ECI**
Cliente	0x1	Id de Series de clientes	**Certificado** de series de Clientes
Series de clientes	0x8	Id de Objetivo de imagen	Imágenes de series de Clientes
Raíz	0x3	Id de Operador, <> 0xFxxxx	**Certificado** de Operador
Raíz	0x3	Id de RL de Operador, == 0xFxxxx	**Lista de Revocación** de Operador
Operador	0x0	Id de Operación de Plataforma, <> 0xFxxxx	**Certificado de Operación de Plataforma**
Operador	0x0	Id de RL de Operación de Plataforma, == 0xFxxxx	RL de **Operación de Plataforma**
Operación de Plataforma	0x0	Id de cofirma de imagen de Operación de Plataforma <> 0xFxxxx	Cofirma de imagen de cliente **Operación de Plataforma**
Operación de Plataforma	0x0	Id de RL de imagen de cliente Operación de Plataforma == 0xFxxxx	**Lista de Revocación de imagen** de cliente **Operación de Plataforma**
Operación de Plataforma o Grupo Objetivo	0x0	Id de Grupo Objetivo, <> 0xFxxxx	Grupo Objetivo, definido en ETSI [UIT-T J.1014]
Operación de Plataforma o Grupo Objetivo	0x0	Id de RL Objetivo, == 0xFxxxx	Lista de Revocación objetivo, definida en [UIT-T J.1014]
Operación de Plataforma o Grupo Objetivo	0x8	Id de Microcliente, <> 0xFxxxx	Microcliente, definido [UIT-T J.1014]
Operación de Plataforma, **Grupo de Exportación**, Grupo de exportación de tercero	0x4	Id de Grupo de Exportación, <> 0xFxxxx	**Grupo de Exportación**
Operación de Plataforma, **Grupo de exportación**, Grupo de exportación de tercero	0x4	Id de RL de Grupo de Exportación, ==0xFxxxx	Lista de Revocación de **Grupo de Exportación**
Grupo de exportación	0x5	Id de Grupo de Exportación de tercero <> 0xFxxxx	**Grupo de Exportación** de tercero
Grupo de exportación	0x8	Id de RL de Grupo de Exportación == 0xFxxxx	Lista de Revocación de **Grupo de Exportación**
Grupo de exportación, Grupo de exportación de tercero	0xE	Id de Sistema de Exportación, <> 0xFxxxx	Sistema de Exportación
Raíz	0x4	Id de Operador con Autorización de Exportación <> 0xFxxxx	Operador con Autorización de Exportación

Cuadro 5.2-2 – Asignación de ID y Padres para elementos con signo

Padre	Tipo	Campo ID	Descripción
Raíz	0x4	Id de Operador con Autorización de Exportación, == 0xFxxxx	Lista de Revocación de Operador con Autorización de Exportación
Operador con autorización de exportación, Autorización de exportación	0x0	Id de Autorización de Exportación, <> 0xFxxxx	Autorización de Exportación (con **Hijos**)
Operador con autorización de exportación, Autorización de exportación	0x0	Id de Autorización de Exportación, == 0xFxxxx	Lista de Revocación de Autorización de Exportación
Otros	Otros		Reservado
NOTA – Las funciones ECI pueden transportar y procesar por separado el campo de **datos** y las secciones de **firma** de un **Certificado** u otro elemento de datos con signo.			

5.3 Lista de Revocación ECI

Una **Lista de Revocación** debe ser firmada por la misma **Entidad** que originalmente firmó el **Certificado** revocado. La **Lista de Revocación** es una lista de identificadores de entidades que definen la versión mínima aceptable de sus **Certificados**. Si una Lista de Revocación contiene un **Certificado** que tiene una o varias listas de revocación asociadas, existe un número de versión mínimo para la **Lista de Revocación** que es aplicable a dicho **Certificado**. En la Figura 5.3-1 se muestra el esquema de una **Lista de Revocación ECI**.

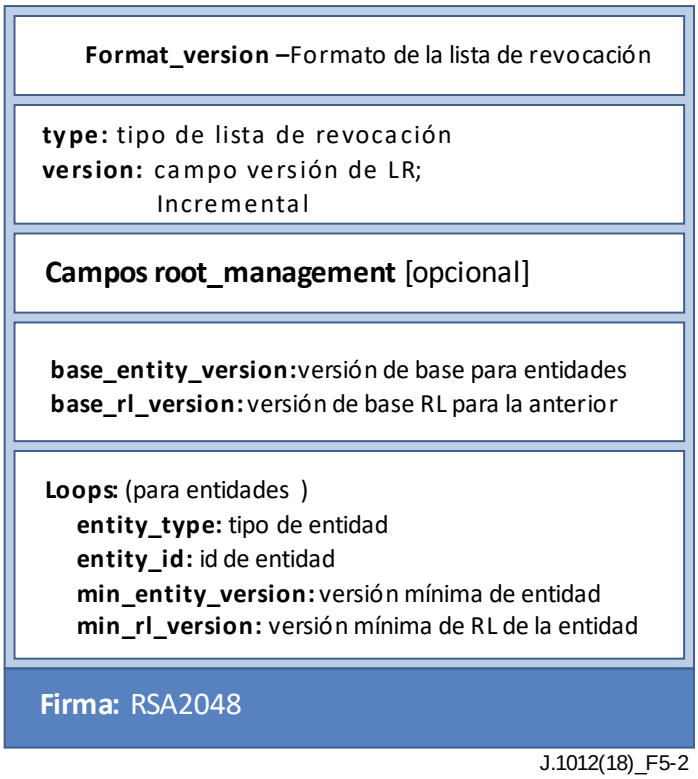


Figura 5.3-1 – Estructura de la Lista de Revocación

Las implementaciones del **Anfitrión ECI** almacenarán la última **Lista de Revocación** recibida (definida en **rl_version**) para cualquier **Entidad** que gestionen, con independencia del origen de los datos.

La **Lista de Revocación** (ECI_RL) se define en el Cuadro 5.3-1.

Cuadro 5.3-1 – Definición de una Lista de Revocación

Sintaxis	N.° de bits	Mnemónico
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 (véase la Nota)	uimsbf
}		
}		
NOTA – = en versión 1 de LRC (listas de revocación de certificado) asociadas al **Certificado**.		

Semántica:

format_version: entero	Valores 0x00, 0x02..0xFF: reservados. Valor 0x01: Versión 1 del formato de la **Lista de Revocación** ECI. Las implementaciones que no reconozcan un tipo de **Certificado** no lo procesarán y devolverán una indicación de fallo a las peticiones de validación.
type: entero	El campo tipo se define en ECI_Certificate_Id, véase el Cuadro 5.3-1.
indicator: entero	Indicación de **Lista de Revocación**; el valor será 0xF.
version: entero	Versión de esta RL. Comienza en 1 (que normalmente está vacío en un **Certificado** nuevo) y se incrementa con cada actualización.
base_entity_version: entero	Se revocan todas las entidades cuyo **id.version** sea inferior a **base_id_version**.
base_rl|_version	Las listas de revocación de una entidad cuya versión **base_entity_version** sea inferior a **base_rl_version** no son válidas.
number_of_entities: entero	Número de entidades en la Lista de Revocación. Véase el Cuadro 5.3-1. Véanse más abajo los valores máximos.
entity_type: entero	Tipo de entidad cuyas versiones más antiguas son revocadas.
entity_id: entero	Entity_id de la entidad cuyas versiones más antiguas son revocadas.
min_entity_version: entero	El número mínimo de versión de la entidad (id de certificado) que concuerda con **entity_type** y **entity_id**. Las versiones inferiores son revocadas.
min_rl_version	Versión mínima de la Lista de Revocación a aplicar conjuntamente con la entidad para la que existe concordancia con **entity_type**, **entity_id** y **entity_min_version**. Las versiones inferiores de Lista de Revocación no son válidas.
root_version_indicator: bit	Si el valor es 0, los campos root_version y min_root_version no serán significativos. Si el valor es 1 y el **Padre** es un **Certificado Raíz** los campos **root_version** y **min_root_version** se interpretarán tal como se indica más abajo.

root_version	Versión del **Certificado Raíz** signatario de esta **Lista de Revocación**.
min_root_version: entero	Si la versión de **Padre** (es decir, la Raíz) es igual o mayor que este campo, todas las versiones del **Certificado Raíz** cuyo valor sea inferior a **min_root_version** serán revocadas para la verificación de **Certificados** del tipo definido en **revocation_id_lead**.
extension: Extension_Field	Datos adicionales: serán ignorados (excepto para el cálculo de la firma) por aquellas implementaciones cuya función no sea interpretar este campo, excepto para el cálculo de la firma.
rl_signature: ECI_Signature_v1	Firma de la **Entidad ECI** a la que se asocia la **Lista de Revocación**. La firma se calcula teniendo en cuenta todos los datos precedentes.

NOTA – Las implementaciones hardware pueden procesar las **Listas de Revocación** por partes, en busca del ID de un **Certificado** posterior que debe validarse al tiempo que se acumula la función hash de la firma y se alcanza el final la lista de verificación de la firma.

Como regla general, los **Anfitriones ECI** almacenarán las **Listas de Revocación de la TA** de todos los **Certificados** necesarios para verificar las entidades que carga el **Anfitrión ECI**. Los **Anfitriones ECI** sustituirán una **Lista de Revocación** almacenada para un **Certificado** o elemento por otra **Lista de Revocación** nueva recibida recientemente cuyo número de versión sea superior.

La máxima longitud de las **Listas de Revocación** será conforme con la cláusula B.2.

5.4 Cadenas de Certificados y Árboles de Listas de Revocación

5.4.1 Definiciones de estructuras de datos

Una **Cadena de Certificados** es una secuencia de **Certificados** con **Listas de Revocación** asociadas, en las que la entidad que gestiona el **Certificado** precedente ha firmado un nuevo **Certificado**. Comienza con la **Lista de Revocación** del **Certificado Padre** (normalmente una raíz). El número de versión mínima (válida) de un **Certificado** y la versión mínima (válida) de la **Lista de Revocación** para un **Hijo** se definen en la **Lista de Revocación** de su **Padre**. Las cadenas se utilizan como credenciales para verificar un elemento a descargar y, por lo tanto, normalmente la **Lista de Revocación** de su predecesor no tendrá un **Certificado**. Sin embargo, es obligatorio procesar la **Lista de Revocación** a fin de verificar la integridad de la cadena. En el Cuadro 5.4.1-1 se representa la estructura de una **Cadena de Certificados** típica.

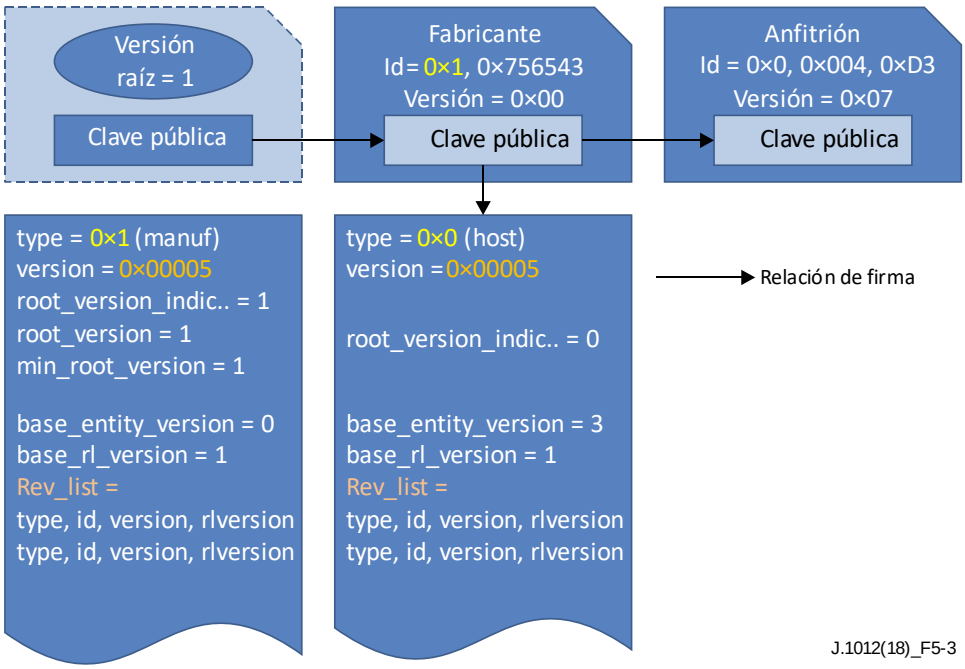


Figura 5.4.1-1 – Ejemplo de Cadena de Certificados de Anfitrión

Las cadenas pueden ser transportadas o almacenadas y componerse a partir de distintas secciones.

Los árboles de listas de revocación son secuencias de listas de revocación vinculadas que usan un **Certificado** de una cadena anterior como **Padre**, abarcando, por lo tanto, una amplia variedad de elementos certificados. Estos pueden ser utilizados por **Operaciones de Plataforma** para descartar por obsolescencia (indicar su revocación) otras entidades (revocadas por la **TA**). La definición de la **Cadena de Certificados** y el árbol de **Lista de Revocación** serán conformes con el Cuadro 5.4.1-1.

Cuadro 5.4.1-1 – Definiciones de Cadena de Certificados y de Árbol de Lista de Revocación

Sintaxis	N.° de bits	Mnemónico
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**		
}		
}		
}		

Semántica:

chain_length: entero	Longitud de la cadena.
rl: ECI_RL	**Lista de Revocación** del **Certificado** o el **Padre** precedente de la cadena en caso de primera iteración de una cadena. Los números de versión del campo identificador de las **Listas de Revocación** en una cadena serán los mismos.
certificate: ECI_Certificate	**Padre** del **Certificado** siguiente en la secuencia actual.
father_revocation_list: ECI_RL	**Lista de Revocación** del **Padre** de la cadena.
three_breadth: entero	Número de subcadenas en el árbol.
father_node_depth: entero	Nivel del **Certificado Padre** en la **Cadena de Certificados** precedente (incluido el **Padre** del árbol). La lista **Padre** heredada es el **Padre** de esta cadena, precedida por su **Padre**, etc. hasta llegar al **Padre** del propio árbol.

Las reglas de ordenación de **Certificados** en los árboles de **Listas de Revocación** son las siguientes:

- Los árboles no contendrán **Certificados** duplicados.
- El árbol se ordenará empezando por la parte más profunda de la estructura topológica (*depth-first*), es decir, todos los **Hermanos** del último **Certificado** hoja se enumerarán como subárboles de chain_lenght=0 inmediatamente después del último **Certificado**, seguidos de los subárboles del **Hermano** del **Padre**, etc.
- Los **Certificados Hermanos** ocuparán en el árbol el orden correspondiente a su identificador (el menor en primer lugar).

5.4.2 Reglas de procesamiento de las Cadenas de Certificados

El **Anfitrión ECI** verifica las **Cadenas de Certificados** y proporciona una contestación adecuada para los elementos revocados utilizando el **Sistema de Seguridad Avanzada**. El **Sistema de Seguridad Avanzada** ejecuta las fases críticas en materia de seguridad de verificación del **Certificado** y de la **Lista de Revocación**. El **Sistema de Seguridad Avanzada** también proporciona a los **Clientes ECI** la capacidad de verificar la validez de los números de versión de revocación de las cadenas.

El **Anfitrión ECI** puede procesar una **Cadena de Certificados** mediante un proceso iterativo. Este comienza con la **Lista de Revocación** Raíz de la **TA ECI** y termina con el último elemento de una cadena. El procesamiento de la **Cadena de Certificados** falla si se produce un error de verificación intermedio. Si el **Anfitrión ECI** falla para una condición dada, asegurará que el **Certificado** y la **Lista de Revocación** actual y todas las **Listas de Revocación** y **Certificados** precedentes son validadas mediante sus firmas antes de activar las medidas políticas del **Anfitrión ECI** sobre entidades revocadas o credenciales inválidas. El **Sistema de Seguridad Avanzada**, tal como se define en [UIT-T J.1014] y [UIT-T J.1015], garantizará que se mantiene un procesamiento robusto adecuado para la **Cadena de Certificados**.

Se permite cualquier orden de procesamiento siempre que produzca el mismo resultado con relación a la aceptación de cadenas:

1) El **Anfitrión ECI** realizará los siguientes pasos de verificación de las **Listas de Revocación**:
   a) El **Anfitrión ECI** verificará si el campo **format_version** de la **Lista de Revocación** corresponde a una versión que pueda interpretar y velará por que los campos **rl_id.type** y **rl_id.rl_indicator** correspondan a los valores previstos.
   b) El **Anfitrión ECI** verificará si la longitud de la **Lista de Revocación** corresponde a sus valores de campos.
   c) Si **root_version_indicator** = 1 el **Anfitrión ECI** verificará si en este punto de procesamiento de la cadena es previsible que una Raíz sea **Padre**, verificará si la **root_version** está disponible para su verificación y comprobará si **min_root_version** no es superior a alguna de las versiones de raíz utilizadas hasta ese momento en el procesamiento de la cadena.
   d) El **Anfitrión ECI** verificará si esta **Lista de Revocación** no ha sido invalidada en función del número de versión mínimo de **Lista de Revocación** de la **Lista de Revocación** anterior en la cadena o en caso de una lista de revocación raíz en función del número min_root_revocation_list utilizado hasta ese momento en el procesamiento de la cadena.
   e) El **Anfitrión ECI** verificará la firma de la **Lista de Revocación** con la clave pública del **Certificado Padre**.
   f) El **Anfitrión ECI** procesará, si tiene capacidad para ello, los bytes de extensión de la **Lista de Revocación**.
   g) El **Anfitrión ECI** verificará si el ***next*** <entity type, entity id, version> de la cadena no es revocado con arreglo a la **Lista de Revocación** y establecerá la versión mínima de la **Lista de Revocación** que debe aplicarse a ese **Certificado**.
2) El **Anfitrión ECI** realizará las siguientes fases de verificación previa del **Certificado** siguiente:
   a) El **Anfitrión ECI** verificará la versión del **Certificado**. Si la versión no corresponde a sus capacidades de procesamiento, no podrá cargar de la cadena.
   b) El **Anfitrión ECI** verificará el campo tipo (*type*) del ID del certificado y se producirá un fallo si este no se corresponde con alguno de los valores esperados.
   c) El **Anfitrión ECI** verificará que la longitud del **Certificado** se corresponde con su definición de formato.

d) El **Anfitrión ECI** verificará la firma del **Certificado** con la clave pública del **Certificado Padre**.

e) El **Anfitrión ECI** procesará, si tiene capacidad para ello, cualquier campo y/o bytes de extensión adicionales del **Certificado**.

La cadena **Lista de Revocación** puede utilizarse, tal como se extrae de un árbol **Lista de Revocación,** para verificar la revocación de un elemento específico que el **Sistema de Seguridad Avanzada** necesita cargar. Ese elemento puede identificarse mediante la secuencia de identificadores de **Certificados** utilizados para verificarlo durante su carga en el **Sistema de Seguridad Avanzada**. Las normas de procesamiento por defecto de una cadena **Lista de Revocación** serán las mismas que las de una **Cadena de Certificados**.

3) El **CPS** cargará la **Lista de Revocación** actual y la <entity type, entity id, version> del **Certificado** siguiente en el CPS. El CPS realizará la siguiente verificación:

a) El CPS comprobará que el campo **format_version** de la **Lista de Revocación** corresponde a una versión que pueda interpretar y que los campos **rl_id.type** y **rl_id.rl_indicator** corresponden a valores esperados.

b) Si el **Padre** es un **Certificado Raíz** (**root_version_indicator**=1) el CPS seleccionará el **Certificado Raíz** con **root_version** como **Padre**, en otro caso, se utiliza el **Certificado** precargado o precedente.

c) El CPS verificará la firma de la **Lista de Revocación** con la clave pública del **Certificado Padre**.

d) El CPS verificará si la longitud de la **Lista de Revocación** corresponde a sus valores de campos.

e) El CPS verificará si el número de versión de la **Lista de Revocación** no ha sido invalidado.

f) El CPS verificará que el ***next*** <entity type, entity id, and version> de la cadena no ha sido revocado con arreglo a la **Lista de Revocación** y establecerá la versión mínima de la **Lista de Revocación** que debe acompañar a dicho **Certificado**.

4) El **Anfitrión ECI** cargará el **Certificado** en la ubicación de procesamiento adecuada del CPS, que realizará las verificaciones siguientes:

a) El CPS verificará que el campo **format_version** de la **Lista de Revocación** corresponde a una versión que pueda interpretar y que los campos **id.type** e **id.entity_id** son valores esperados.

b) El CPS verificará si la longitud del **Certificado** corresponde con sus valores del campo.

c) El CPS verificará la firma con arreglo a la clave pública del **Certificado Padre**.

5.5 Conjuntos de árbol de revocación y ficheros de datos de revocación

Los datos de revocación para verificar una **Entidad** específica deben seleccionar datos de revocación que contienen la RL (Lista de Revocación) del **Padre** de la **Entidad** objetivo.

Cuando se distribuyen datos de revocación, las cadenas para revocar múltiples entidades objetivo pueden combinarse en un árbol, evitando así la duplicación de la Raíz y de **Certificados Hijo** así como de sus **Listas de Revocación** y permitiendo una búsqueda más ordenada en los **CPE**.

Para facilitar la agrupación y desagrupación de datos de revocación, los árboles de revocación pueden simplemente combinarse en un conjunto de árboles. No obstante, no habrá solapamiento entre conjuntos de árboles excepto en lo relativo a la lista de revocación del **Padre** común. Los conjuntos de árboles pueden contener múltiples RL Raíz (durante un despliegue de cambio Raíz de **TA**).

La definición de la **Cadena de Certificados** y del árbol **Lista de Revocación** será conforme con lo indicado en el Cuadro 5.5-1**.**

Cuadro 5.5-1 – Definición del conjunto Árbol de Lista de Revocación

Sintaxis	N.° de bits	Mnemónico
ECI_RL_Tree_Set {		
tree_number	32	uimsbf
for (i=0; i<tree_number; i++) {		
ECI_RL_Tree **tree**		
}	8	uimsbf
}		

Semántica:

tree_number: entero	Número de árboles en el conjunto.
tree: ECI_RL_Tree	Árbol (incluido el **Certificado Raíz**) de **Certificados** y sus **Listas de Revocación**.

NOTA – Los servidores en línea pueden distribuir árboles de una única **Entidad** objetivo (efectivamente cadenas) para minimizar el tráfico de datos. En las redes de difusión los árboles pueden dividirse y fusionarse fácilmente para lograr la concordancia en el número de colectores (véase 7.7.2) utilizados en el carrusel de transmisión.

No es necesario que los árboles o conjuntos de árboles de revocación sean completos en el sentido de contener todas las entidades de una clase. La **Operación de Plataforma** es responsable de establecer el conjunto de árboles de revocación como considere mejor adaptado, minimizando el riesgo en los **CPE** desplegados en la red de la **Operación de Plataforma**. En redes de difusión, las **Listas de Revocación** pueden alternarse en el tiempo a fin de ampliar la cobertura de la revocación.

La **ECI** requiere que los **CPE** almacenen de forma permanente cadenas de **TA ECI** para todos los elementos que puedan cargarse a fin de garantizar que las entidades que han sido revocadas queden efectivamente revocadas de forma permanente. Esto se especifica en las cláusulas pertinentes de este documento.

Por conveniencia en el transporte, los conjuntos de árboles de revocación **ECI** se transportan con el formato que figura en el Cuadro 5.5-2.

Cuadro 5.5-2 – Fichero de datos de revocación

Sintaxis	N.° de bits	Mnemónico
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**		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para verificar el formato de los datos que siguen. Toma el valor de los tres caracteres ASCII de 8 bits 'ERD'. El **Anfitrión ECI** verificará el valor de este campo para comprobar si un fichero **ECI** tiene el formato previsto a los efectos de la integridad de datos adicionales.
version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida, los restantes valores están reservados. El **Anfitrión ECI** ignorará cualquier imagen cuyo número de versión no sea reconocible.
father_type: entero	Tipo del **Padre** común de los datos de la **Lista de Revocación**. El valor 0x0 indica el **Certificado Raíz ECI**. Los valores 0x1 a 0x7 están reservados. Los valores 0x8 a 0xF pueden utilizarse en aplicaciones privadas.
sub_type: entero	Si el campo father_type es 0x0, el tipo de **Lista de Revocación** común de define con arreglo al Cuadro 5.2-2 del **Certificado Raíz ECI** de los datos contenidos en los datos de revocación. Este valor no está definido para otros valores de father_type.
revocation_data: ECI_RL_Tree_Set	Conjunto de Árbol Lista de Revocación de listas de revocación para elementos revocados.

5.6 Firmas de elementos de datos de gran tamaño

La **ECI** calcula las firmas para elementos de datos de gran tamaño (por ejemplo, software de imágenes) utilizando funciones hash eficientes para su aplicación a datos masivos conjuntamente con una operación de firma ordinaria. En el Cuadro 5.6-1 se define la firma de elementos de datos de gran tamaño.

Cuadro 5.6-1 – Definición de la firma de elementos de datos de gran tamaño

Sintaxis	N.° de bits	Mnemónico
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
}		
}		
}		

Semántica:

sign_version: entero	Versión de la firma. El valor 0x01 corresponde a la versión actual; los demás valores de versión están reservados. Los **CPE** que no hayan implementado una versión ignorarán este campo (y cualquier dato que siga).
signature_byte: byte	Secuencia de bytes que representa la firma del elemento de gran tamaño.

El algoritmo de firma se define en el Anexo A.

5.7 Certificados Raíz

5.7.1 Definición del Certificado Raíz

La **ECI** utiliza una secuencia de ***versiones*** de **Certificados Raíz**. La **TA ECI** puede iniciar el uso de una nueva versión de **Certificado Raíz**, por ejemplo, cuando cualquiera de las **Listas de Revocación** anteriores de cualquiera de los **Hijos** sea demasiado grande o si se considera que la clave secreta asociada con la clave pública del **Certificado** ha dejado de ser suficientemente secreta.

Un **Certificado Raíz** utiliza el campo identificador de **Certificados ECI** con la definición de campo que figura en el Cuadro 5.7-1. Los campos tipo e identificador nunca se utilizan; sólo se aplica el campo versión.

Cuadro 5.7-1 – Definición del campo Root_ID ECI

Sintaxis	N.° de bits	Mnemónico
ECI_Root_Id {		
type /* véase el Cuadro 5.2-1***/**	4	uimsbf
id /* véase el Cuadro 5.2-2***/**	20	uimsbf
version	8	uimsbf
}		

Semántica:

version: entero	Número de versión del **Certificado**; la numeración comienza por 1 y se incrementa de uno en uno con cada nuevo **Certificado Raíz** generado. El valor 0x00 está reservado.

5.7.2 Gestión del Certificado Raíz del Anfitrión ECI

La **TA ECI** puede iniciar el uso de un nuevo **Certificado Raíz** con un número de versión más alto. El algún momento posterior, podrá emitir una **Lista de Revocación** para el nuevo **Certificado Raíz** que revoque los **Certificados Raíz** precedentes. Ello invalida todos los **Certificados** firmados por dicha Raíz.

Alternativamente, la **TA ECI** puede decidir que una **Lista de Revocación** para tipos específicos de entidades (por ejemplo, **Fabricantes**) es demasiado amplia y, en consecuencia, vuelve a emitir nuevas versiones de todos los **Certificados** previamente emitidos utilizando un campo **min_id_version** en la **Lista de Revocación** para ese tipo de **Entidad** que tenga un valor mayor. Ello invalida efectivamente todos los **Certificados** anteriormente emitidos para el tipo de Entidad hasta el valor **min_entity_version**-1. Ello requiere por lo general emitir numerosos nuevos **Certificados** con un número de versión más alto para entidades que aún utilicen una versión inferior del **Certificado** para sustituir los **Certificados** revocados.

En [b-UIT-T J Supl. 7] se definen los recursos que un **Anfitrión ECI** proporcionará para el almacenamiento de **Certificados Raíz**.

6 Cargador de Anfitrión ECI

6.1 Introducción

En el proceso de carga del **Anfitrión ECI** se distinguen los aspectos siguientes:

1) El almacenamiento de una imagen, la verificación de la autenticidad de la imagen por el **CPE** utilizando los datos de autenticación proporcionados por la **TA ECI** y la subsiguiente activación de la imagen.
2) El formato de fichero del fichero o ficheros que contienen la imagen y el resto de información necesaria para cargar la imagen en el **CPE**.
3) El protocolo de transporte para entregar la **Imagen de Anfitrión ECI** al **CPE**. Ello incluye cualquier descubrimiento que realice el **CPE** sobre la ubicación de las imágenes necesarias. Incluye asimismo cualquier almacenamiento de las imágenes transportadas, la cadena de validación **ECI** y los datos de firma complementarios.
4) Cualquier revocación específica del **Operador** de **Imágenes de Anfitrión ECI**; el formato de los datos de dicha información se define en la cláusula 6; su aplicación se define en la cláusula 8.

La lógica de la verificación y la autenticación de la imagen se aplicará a nuevas **Imágenes de Anfitrión ECI** y datos de autenticación descargados, en cada reinicialización del software de un **CPE** y cuando se provisione durante el funcionamiento normal de un **CPE**.

6.2 Almacenamiento, verificación y activación

6.2.1 Principios de operación

El **Anfitrión ECI** garantiza que los **Clientes ECI** puedan ejecutarse en un entorno privado y sin manipulación, con arreglo a los requisitos de robustez de la **ECI** para la implementación de dichos clientes. El **Anfitrión ECI** también impide la interferencia entre dos **Clientes ECI**. A tal fin, la **TA ECI** puede certificar software para los **CPE** y el cargador del **CPE** verificará la autenticidad de las imágenes software que carga.

Muchos **CPE** utilizan cargadores multietapa. La **ECI** asume que el chip principal del **CPE** carga un conjunto de imágenes de inicialización específicas para el chip antes de comenzar la carga de imágenes software ordinarias. Dichas imágenes pueden ser certificadas implícitamente en el marco del acuerdo de licencia del suministrador del chip con la **TA ECI**. Alternativamente, pueden formar parte del proceso de certificación del **Fabricante** que se define en esta cláusula.

Si posteriormente se demuestra que el software de alguna de las imágenes gestionadas por la **ECI** tiene un fallo de seguridad, la **TA ECI** y el **Fabricante del CPE** pueden revocarla y sustituirla por una versión en la que se haya reparado el fallo.

En la Figura 6.2.1-1 se asume que **Img1** es una imagen específica de un chip que es necesaria para que el chip pase a un estado que le permita iniciar la carga de imágenes de aplicaciones adicionales ordinarias. Está protegida por una firma específica de chip, **CS1,** que es verificada por el **Cargador del Chip** mediante una clave propiedad del suministrador del chip.

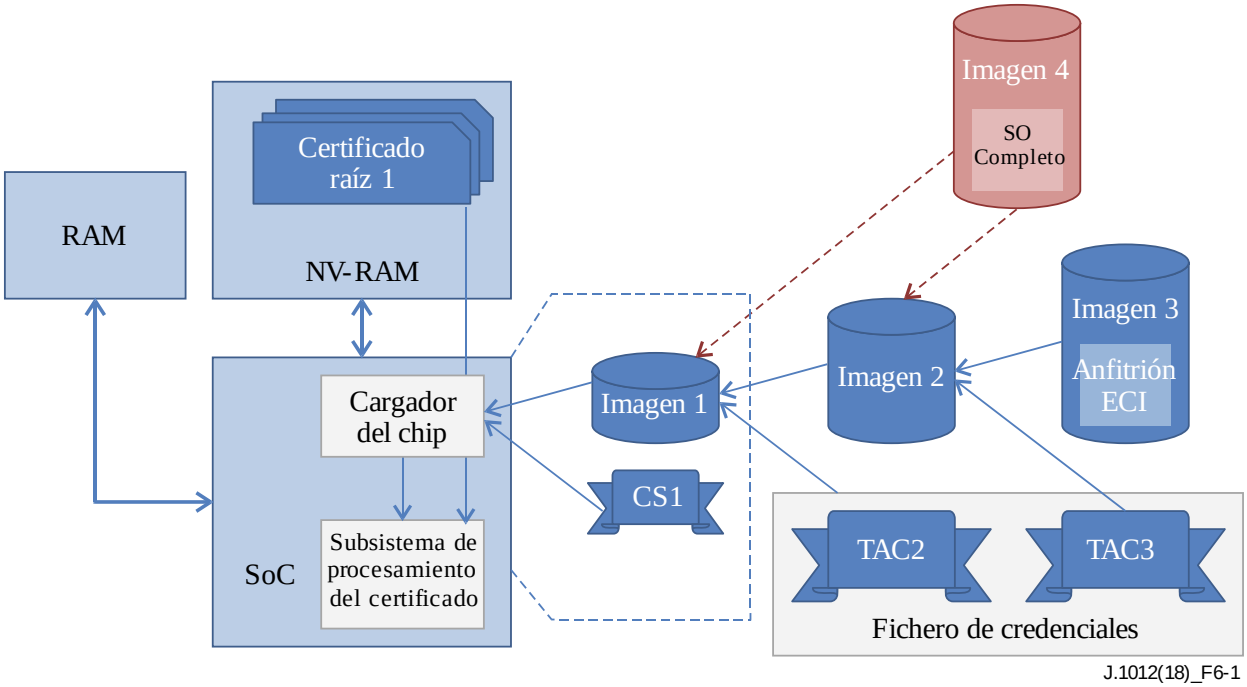


Figura 6.2.1-1 – Ejemplo de proceso de carga del Anfitrión ECI

Una vez que **Img1** se ejecuta, el chip procede a cargar otras imágenes. Carga **Img2**, que puede ser autenticada por una **Cadena de Certificados** y la **TAC2** de la firma de la imagen. La imagen se verifica utilizando el **Certificado Raíz TA**, el **Subsistema de Procesamiento del Certificado** y la **TAC2**. **Img2** carga entonces **Img3**, que contiene el software del **Anfitrión ECI**. **Img3** se verifica mediante el **Subsistema de Procesamiento del Certificado,** los **Certificados Raíz** y la **Cadena de Certificados** de la Autoridad de Confianza y la **TAC3** de la firma de la imagen. Otras imágenes adicionales como **Img4** que contiene, por ejemplo, un OS completo pueden cargarse sin estar certificadas por la **TA ECI** si el entorno de la carga puede asegurar que ello no supone un peligro para la seguridad del **Anfitrión ECI**.

Las Credenciales de la Autoridad de Confianza (TAC) de las imágenes se transportan en un fichero especial de credenciales.

La **TA ECI** certifica la integridad de la seguridad del **Anfitrión ECI**: su capacidad de proporcionar privacidad al cliente, resistir las manipulaciones externas al **Anfitrión ECI** y asegurar a los clientes que no se producirán interferencias entre ellos. Los **Fabricantes de CPE** pueden utilizar medidas de seguridad complementarias para la carga de imágenes utilizando sus mecanismos patentados de encriptación y autenticación.

Las operaciones de plataforma pueden verificar el carácter actualizado de las **Imágenes del Anfitrión ECI** y decidir no desencriptar determinados servicios. A tal fin, el **CPS** extrae el número más bajo de versión de **Lista de Revocación** utilizada para verificar cualquier elemento cargado, permitiendo así que las **Operaciones de Plataforma** verifiquen la posible aplicación de un **Lista de Revocación** reciente. En la cláusula 8 se definen estos procedimientos de aceptación específicos de la **Operación de Plataforma** para un **Anfitrión ECI**.

El **Cargador del Anfitrión ECI** almacenará en NV-RAM las últimas **Imágenes del Anfitrión ECI** y las credenciales más recientes. El **Cargador del Anfitrión ECI** verificará de nuevo todas las imágenes cargadas cada vez que se reinicialice el software del **Anfitrión ECI**. Este procedimiento establece la autenticidad del **Anfitrión ECI** en cada reinicio del software.

6.2.2 Definición de credenciales

6.2.2.1 Certificados relacionados con las Imágenes del Anfitrión ECI

La **ECI** proporciona dos tipos de **CPE ECI** en lo que se refiere a la diversidad de las **Imágenes de Anfitrión ECI**:

1) **CPE** genéricos que cargarán el mismo conjunto de **Imágenes de Anfitrión ECI** en cada instancia del mismo tipo del **CPE** y versión.
2) **CPE** individualizados que cargarán (parcialmente) un conjunto de imágenes diferentes en cada instancia del mismo tipo de **CPE** y versión**.** Dichas series de imágenes del mismo "tipo" pero individualizadas para cada **CPE** se denominan **Series de Imágenes**.

La **Cadena de Certificados del Anfitrión ECI** consta de los **Certificados** siguientes (cada uno certificado por su predecesor):

1) Certificado Raíz:
   – Es la representación de la **Entidad** Raíz de la **TA ECI**. La clave pública de este **Certificado** se utilizará para la verificación.
2) Certificado de Fabricante:
   – Es una representación de la **Entidad TA ECI** para un **Fabricante** específico. La clave pública de este **Certificado** se utilizará para la verificación.
3) Certificado del Anfitrión:
   – Es una representación del hardware de un **CPE** certificado por la **TA ECI** y de la versión del software del **Anfitrión ECI**. En el caso de **Anfitriones ECI** genéricos, la clave pública de este **Certificado** se utilizará para autenticar todas las **Imágenes de Anfitrión ECI**. La clave pública de las **Imágenes de Anfitrión ECI** "individualizadas" se utilizará para la verificación.
4) Certificado de Series de Imágenes de Anfitrión:
   – Esta **Entidad** proporciona una aprobación genérica para una serie de imágenes específicas de una configuración específica del **CPE**, pero que por lo demás son idénticas desde la perspectiva de una **TA ECI.** ***En el caso de*** **Anfitriones ECI** individualizados la clave pública de este **Certificado** se utilizará para autenticar la **Imagen de Anfitrión ECI** destinada a un **CPE** específico con un ID de **CPE** que concuerde con el identificador incluido en el **Certificado**.

NOTA – Cada identificador de Entidad debe interpretarse en el contexto de la Entidad que autoriza; es decir, los ID son relativos.

La **Imagen de Anfitrión ECI** y la estructura de certificación conexa se describen de forma esquemática en la Figura 6.2.2.1-1 y el Cuadro 6.2.2.1-1 y ofrece una visión general de los parámetros conexos.

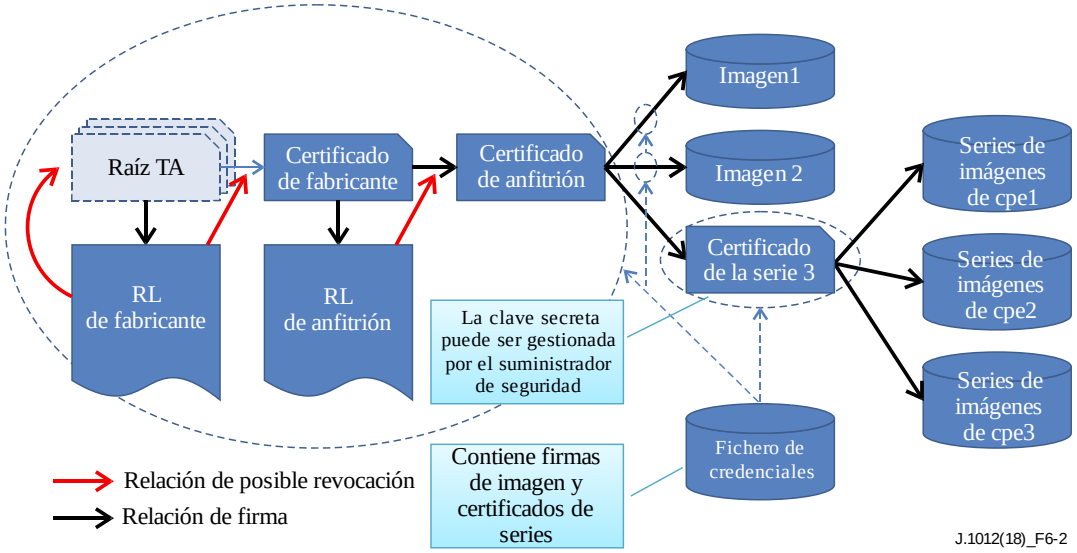


Figura 6.2.2.1-1 – Estructura de la certificación de la Imagen de Anfitrión ECI

Cuadro 6.2.2.1-1 – Visión general de los parámetros de Certificados asociados al Anfitrión ECI

Tipo	Entidad	Valor del campo ID de certificado	Procesamiento específico del Anfitrión ECI
0x0	**Fabricante**	manufacturer_id, version	Se verifica el Manutacturer_id con relación al ID de fabricante del **CPE** en el bloque de AS (seguridad avanzada).
0x0	Anfitrión	cpe_type, cpe_model, host_version	cpe_type y cope_model se verifican con relación al tipo de **CPE** y el modelo de **CPE** en el bloque de AS
0x8	**Series de Imágenes** de CPE	target_id	Se verificará el target_id con relación a la identidad del **CPE**.
0x8	Imagen de CPE	N.A.	
0x8	**Imagen de Anfitrión ECI**	ECI_Host_Image_Id	Este es el tipo para la firma real de la imagen.

Las definiciones de **Certificados** para los **Certificados** asociados al **Anfitrión ECI** serán conformes con la definición de **ECI_Certificate** de la cláusula 5.2. La definición de los campos identificadores de los **Certificados** para la gestión del **Anfitrión ECI** se presenta en el Cuadro 6.2.2.1-2.

Cuadro 6.2.2.1-2 – Definición del campo ID de Certificados asociados al Anfitrión

Sintaxis	N.° de bits	Mnemónico
ECI_Manufacturer_Id {		
padding(4)		
type /* véase el Cuadro 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 /* véase el Cuadro 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 /* véase el Cuadro 5.2-2 */	4	uimsbf
image_series_model	8	uimsbf
image_series_model_extension	4	uimsbf
image_series_version	16	uimsbf
}		

Semántica:

type	Valor conforme al Cuadro 5.2-2.
manufacturer_id: entero	Id asignado al **Fabricante** por la **TA ECI**.
cpe_type: entero	Id asignado al modelo de **CPE** por el TA **ECI**. Los valores 0x000 y 0x3F0..0x3FF están reservados. Los **CPE** del mismo modelo tendrán numerosos aspectos en común y utilizarán la misma tecnología de seguridad **ECI**.
cpe_model: e	Id asignado a una versión de un modelo específico que en muchos aspectos es idéntico, pero con algunas diferencias que no son triviales. La **TA ECI** asigna los valores, estando reservados 0x00 y 0xF0..0xFF.
cpe_type_id: ECI_CPE_Type_id	ID del tipo de hardware del **CPE** (versión + modelo); es único en el contexto del manufacturer_id.
cpe_host_version	ID asignado a un conjunto de imágenes que conforman una configuración de **Anfitrión ECI de CPE** para el **CPE**.
image_series_model: entero	ID de imágenes del mismo tipo para los **CPE** que soportan las **Series de Imágenes**, viniendo la distinción dada por el **cpe_id**. Los valores 0x000 y 0xF00..0xFFF están reservados.
image_series_version: entero	Id asignados de forma incremental a la versión del modelo de **Series de Imágenes** por la TA ECI. Los valores 0x0000 y 0xF000..0xFFFF están reservados.

6.2.2.2 Firmas de Imágenes del Anfitrión ECI

El ID de la **Imagen de Anfitrión ECI** coincidirá con el id de Series de Imágenes del Anfitrión, y se define en el Cuadro 6.2.2.2-1.

Cuadro 6.2.2.2-1 – Definiciones del ID de la Imagen de anfitrión y del ID de Series de Imágenes del Anfitrión

Sintaxis	N.° de bits	Mnemónico
ECI_Host_Image_Id {		
padding(4)		
type /* véase el Cuadro 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
}		
}		

Semántica:

type	Valor conforme al Cuadro 5.2-2.
image_model: entero	Id asignado a una **Imagen de Anfitrión ECI** o a series de imágenes que se van sustituyendo entre sí. Los valores 0x00 y 0xF0..0xFF están reservados.
image_model_extension: entero	Ampliación del campo anterior. En aplicaciones ordinarias este campo debería fijarse en 0x0.
image_version: entero	Versión de una imagen del mismo tipo asignada de forma incremental. Los valores 0x00 y 0xF0..0xFF están reservados.
cpe_serial_number: entero	Número de serie del **CPE** al que está destinada la imagen. El cpe_serial_number será exclusivo en el contexto de <manufacturer_id, cpe_type_id>.
cpe_type: entero	Campo cpe_type como se define en la estructura de ECI_CPE_Type_Id.
manufacturer_id: entero	Campo manufacturer_id como se define en la estructura de ECI_Manufacturer_Id.
target type: entero	Tipo de identificación objetivo para las series de imágenes. El valor 0x1 define la estructura e indica que se utiliza un cpe_id como objetivo, el resto de valores están reservados.
cpe-id: ECI_CPE_Id	ID del **CPE** que es el objetivo de una serie de imágenes (de **Anfitrión ECI** o de **Cliente ECI**).

Las firmas de las **Imágenes de Anfitrión ECI** y de las **Series de Imágenes de Anfitrión ECI** utilizadas para la firma de las **Imágenes de Anfitrión ECI** reales utilizarán la estructura de firma de datos de gran tamaño que se define en la cláusula 5.5.

6.2.2.3 Credenciales del Anfitrión ECI

El Cuadro 6.2.2.3-1 define la estructura de credenciales del **Anfitrión ECI** que verifica un conjunto de **Imágenes de Anfitrión ECI**.

Cuadro 6.2.2.3-1 – Definición de la estructura de credenciales del Anfitrión ECI

Sintaxis	N.° de bits	Mnemónico
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**		
}		
}		

Semántica:

image_credential_version: byte	Versión del formato de las credenciales. El valor 0x01 es la versión actualmente definida; los demás valores están reservados. Los **Cargadores del Anfitrión ECI** ignorarán cualquier credencial cuyo valor no reconozcan.
image_chain: ECI_Certificate_Chain	**Cadena de Certificados** de 2 niveles que comienza con la **RL Raíz del Fabricante** hasta el **Certificado de Anfitrión ECI**. El último **Certificado** se utilizará para verificar la firma de la imagen de cualquier **Certificado de Series de Imágenes**.
nr_images: entero	Número de imágenes para las que se incluyen las firmas.
image_id	ID de la imagen cuya firma es la siguiente en el bucle. Los identificadores de imagen (image_id) enumerados en el bucle tendrán valores distintos del campo **image_id.image_model**.
series_cert: ECI_Certificate	Certificado utilizado para la verificación de un **Serie de Imágenes**
image_signature: ECI_Data_Signature	Firma de la imagen (incluido el Id de Imagen de Anfitrión).
extension: Extension-Field	Campo de ampliación de retrocompatibilidad.

Cuando el **CPE** verifica la **image_chain**, seguirá las normas de procesamiento genéricas para cadenas definidas en la cláusula 5.4.

6.2.3 Proceso de carga de ficheros Imagen de Anfitrión ECI

El **CPE** almacenará, verificará y activará la ejecución del conjunto de ficheros **Imagen de Anfitrión ECI** necesarios para arrancar el **Anfitrión ECI**. La activación real de la **Imagen de Anfitrión ECI** ocurre normalmente durante la carga inicial del CPE.

El CPE utilizará una función de procesamiento robusta denominada **Cargador de Anfitrión ECI** para descargar, verificar y activar la **Imagen de Anfitrión ECI** elegida. Por ejemplo, si la imagen de carga inicial del **CPE** que contiene el **Cargador de Anfitrión ECI** arranca la ejecución de una segunda imagen, y la segunda imagen carga y arranca la ejecución de una tercera imagen, la funcionalidad de la segunda imagen de cargar adecuadamente la tercera verificación del funcionamiento de la firma de imagen se considerará una funcionalidad del **Cargador de Anfitrión ECI** para ese **CPE**. Sólo la función **Cargador de Anfitrión ECI** puede verificar y arrancar una

Imagen de Anfitrión ECI. El **Cargador d**e **Anfitrión ECI** utilizará el **Sistema de Procesamiento de Certificados** (CPS) para verificar las credenciales de la imagen.

El **CPE** almacenará el último conjunto de ficheros de **Imagen de Anfitrión ECI** y sus credenciales que ha descargado en la **memoria NV**. Durante la carga inicial del **CPE** el **Cargador de Anfitrión ECI** podrá localizar dichos ficheros y comenzar la carga de las imágenes de una forma adecuada para el tipo específico de **CPE**.

El **Cargador de Anfitrión ECI** aplicará, utilizando el CPS, las normas de procesamiento ordinarias indicadas en 5.4 para verificar cada una de las imágenes cargadas. Las imágenes genéricas y los **Certificados de Series de Imágenes** se verificarán utilizando la clave pública del **Certificado** del Anfitrión. La clave pública del **Certificado de Series de Imágenes** se utilizarán para verificar las imágenes de las **Series de Imágenes** y el **CPE** comparará el cpe_id de la imagen con el cpe_id del CPE.

En caso de que una imagen esté en riesgo (fallo de verificación de firma por el CPS), el **Cargador de Anfitrión ECI** rechazará la imagen, es decir, el CPE no podrá instanciar un **Anfitrión ECI** en el **CPE**. El **CPE** podrá recuperarse de esta situación gracias a un procedimiento de recuperación que le permite reinicializar la última **Imagen de Anfitrión ECI** y sus credenciales, por ejemplo, mediante la recarga del último conjunto de ficheros **Imágenes de Anfitrión ECI** desde el canal de difusión, desde su servidor de **Imágenes de Anfitrión ECI** o mediante cualquier otro medio.

El **Anfitrión ECI** almacenará las últimas versiones de los **Certificados** de la cadena del **Anfitrión ECI** que adquiera con independencia del canal a través del cual hayan sido adquiridas. Ello "fija" efectivamente el último **Certificado** de anfitrión disponible como base de futuras verificaciones de imágenes.

La secuencia de carga de las **Imágenes de Anfitrión ECI** no se verifica directamente mediante el proceso de verificación de la firma: se realizará mediante el cargador del software de inicialización de la primera **Imagen de Anfitrión ECI** y para activaciones posteriores mediante las **Imágenes de Anfitrión ECI** precedentes.

6.3 Formatos de ficheros conexos del Anfitrión ECI

En la presente Recomendación no se define ninguna denominación de fichero u otros metaatributos de los ficheros **Imagen de Anfitrión ECI**. Los datos de la **Imagen de Anfitrión ECI** se gestionan como un conjunto de contenedores de datos (ficheros sin nombre en toda la **ECI**) identificados mediante su identificador de Imagen de Anfitrión y las credenciales **ECI** (**Cadena de Certificados** y firmas) necesarias para autenticarlos.

Un fichero **Imagen de Anfitrión ECI** es una secuencia compuesta de un ECI_Host_Image_Header y el contenido de la imagen. Será conforme con la definición que figura en el Cuadro 6.3-1.

Cuadro 6.3-1 – Definición del fichero de Imagen de Anfitrión ECI

Sintaxis	N.° de bits	Mnemónico
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
}		
}		
}		

Semántica:

host_image_byte: byte	**Imagen de Anfitrión ECI** real; formato patentado para el **CPE**.
magic: byte[3]	El número mágico se utiliza para verificar el formato de los datos que siguen. Toma el valor ASCI de los tres caracteres de 8 bits `'EHI'`. El firmware del **CPE** verificará el valor de este campo para comprobar si un fichero **ECI** tiene el formato previsto para la integridad de datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida; los restantes valores están reservados.
host_image_id: ECI_Host_Image_Id	ID de **Imagen de Anfitrión ECI** de la imagen. Los **CPE** verificarán este campo antes de cargar una (nueva) **Imagen de Anfitrión ECI**.
manufacturer_id: ECI_Manufacturer_Id	ID_Manufacturer (ID de fabricante) de la ECI del **Fabricante** del **CPE** de la **Imagen de Anfitrión ECI**. Los **CPE** verificarán este campo antes de cargar una (nueva) **Imagen de Anfitrión ECI**. Véase la Nota.
extensions: Extension_Field	Véase la cláusula 5.1 de la presente Recomendación: ampliaciones retrocompatibles.
host_image_byte: byte	**Imagen de Anfitrión ECI** real.
NOTA – Este también debería corresponder con el OUI del **Fabricante** utilizado en carruseles de difusión para transportar el fichero asociado.	

Los ficheros de **Series de Imágenes** tienen una firma única que se transporta en el propio fichero de imagen. Por lo tanto, el formato específico del fichero seguirá la definición que figura en el Cuadro 6.3-2.

Cuadro 6.3-2 – Definición del fichero de Series de Imágenes de Anfitrión ECI

Sintaxis	N.° de bits	Mnemónico
`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
`    }`		
`  }`		
`}`		

Semántica:

host_image_byte: byte	**Imagen de Anfitrión ECI** real; formato patentado para el **CPE**.
magic: byte[10]	Número mágico utilizado para verificar el formato de los datos que siguen. Toma el valor ASCI de los tres caracteres de 8 bits `'EHS'`. El firmware del **CPE** verificará el valor de este campo para comprobar si existe un fichero ECI con el formato previsto para la integridad de datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida; los restantes valores están reservados.
image_signature: ECI_Data_Signature	Firma para todos los datos del fichero imagen que sigue.
target_id: ECI_Series_Image_Target_Id	ID Objetivo de la imagen. El valor de target_id.target_type es 0x01, los restantes valores están reservados.
extensions: Extension_Field	Véase la cláusula 5.1 del presente documento: ampliaciones retrocompatibles.
host_image_byte: byte	Secuencia de bytes que forman la **Imagen de anfitrión**.

Las credenciales de la **Imagen de Anfitrión ECI** son conformes con la definición del Cuadro 6.3-3, que básicamente es la **Cadena de Certificados** con el conjunto de firmas de imagen o de **Certificados de Series de Imágenes**.

Cuadro 6.3-3 – Definición del fichero de credenciales de la Imagen de Anfitrión ECI

Sintaxis	N.° de bits	Mnemónico
ECI_Host_Image_Credential_File{		
magic = 'EHC'	24	uimsbf
version	8	uimsbf
if (version == 0x01) {		
ECI_Host_Credentials **credentials**		
}		
}		

Semántica:

magic	Número mágico utilizado para verificar el formato de los datos que siguen. Toma el valor ASCI de los tres caracteres de 8 bits 'EHC'. El firmware del **CPE** verificará el valor de este campo para comprobar si algún fichero **ECI** tiene el formato previsto para la integridad de datos adicionales.
version	Versión del formato del fichero. El valor 0x01 corresponde a la versión actualmente definida; los restantes valores están reservados.
credentials: ECI_Host_Credentials	Credenciales para una **Imagen de Anfitrión ECI** o para un grupo de imágenes.

El host_image_id se utiliza para identificar firmas de **TA ECI** para un conjunto de ficheros de **Imágenes de Anfitrión ECI** que incluyen una descarga completa en la estructura de credenciales ECI.

Los **CPE** conformes con la **ECI** pueden descargar otros módulos de software del CPE patentados utilizando el mismo protocolo de transporte que el usado para los ficheros **Imagen de Anfitrión ECI**. Dichas imágenes no requieren un formato específico.

En el medio de difusión es conveniente distribuir los datos de revocación de numerosos **Anfitriones ECI** como un único fichero de gran tamaño. Los **Anfitriones ECI** que reciben esos datos pueden utilizarlos para comprobar sus propios **Certificados de Anfitrión ECI**.

El fichero de datos de revocación del **Anfitrión ECI** utiliza el formato ECI_Revocation_Data_File definido en el Cuadro 5.5-2. El fichero de datos de revocación **Anfitrión ECI** utiliza un father_type igual a 0x0 (**Certificado Raíz**) y un sub_type igual al tipo Lista de Revocación del **Fabricante**. Los datos de revocación (revocation_data) cumplen la restricción de que las listas de revocación hoja en los árboles sean listas de revocación del **Anfitrión ECI**.

6.4 Protocolos de transporte de Imagen de Anfitrión ECI

6.4.1 Introducción

En la presente Recomendación se distinguen tres formas de distribución de ficheros **Imagen** de Anfitrión:

1) **Difusión**: La **ECI** define protocolos para permitir a los **Operadores de Plataformas** señalizar y distribuir nuevos ficheros **Imagen de Anfitrión ECI** desde los **Fabricantes de CPE** a los **CPE** instalados utilizando DVB-SSU.

2) **En línea**: La **ECI** permite que **CPE** conectados a Internet descarguen ficheros **Imagen de Anfitrión ECI** utilizando cualquier protocolo propiedad del fabricante, para lo que se sugiere utilizar HTTP 1.1 así como una interfaz definida por la **ECI** con un servidor en Internet de un operador.

3) **Otras**: Los **Fabricantes de CPE** y/o los **Operadores** también pueden utilizar otros medios para distribuir ficheros **Imagen de Anfitrión ECI**, incluyendo métodos fuera de línea tales como la distribución mediante llaveros USB. Este medio de transporte de imágenes queda fuera del alcance de esta Recomendación. Sin embargo, las imágenes cargadas mediante ese protocolo deberán ser conformes con el formato del fichero y la verificación de la imagen descritas en las cláusulas 6.2 y 6.3.

Los **CPE** diseñados para adquirir **Servicios** de redes de difusión digitales implementarán el protocolo de transporte de difusión de la **Imagen de Anfitrión ECI** definido en 6.4.2.

Los **CPE** con conexión IP implementarán el protocolo en línea de transporte por internet de la **Imagen de Anfitrión ECI** definido en 6.4.3 así como el protocolo definido en 7.7.3.3.

Un **CPE** puede implementar cualquier protocolo de transporte de la **Imagen de Anfitrión ECI** incluidos los protocolos de difusión y de transporte fuera de línea del **Anfitrión ECI** (por ejemplo, con un llavero USB). En todos los casos, el **Fabricante del CPE** asegurará los medios prácticos necesarios para la actualización del **Anfitrión ECI** sobre el terreno mediante una combinación de los anteriores protocolos de transporte, teniendo en cuenta casos prácticos de uso en los que algunas de las conexiones de red no están disponibles.

6.4.2 Protocolo de transporte de difusión del Anfitrión ECI

6.4.2.1 Generalidades y establecimiento del perfil

El protocolo de transporte de difusión del **Anfitrión ECI** permite el transporte de nuevos ficheros **Imagen de Anfitrión ECI** y de los datos conexos desde el **Fabricante del CPE** desde la infraestructura de la cabecera de difusión del **Operador** hasta el **CPE**. El protocolo también permite el transporte de ficheros que no sean **Imagen de Anfitrión ECI** (para funciones no críticas desde el punto de vista de la seguridad). El **Operador** puede jugar un papel activo en la gestión de la versión del software instalado en el **CPE**. Este protocolo facilita la colaboración al establecer normativa para los puntos de interoperabilidad técnica entre el **Fabricante del CPE** y el **Operador**:

- Traspaso voluntario normalizado de datos de descarga desde el **Fabricante del CPE** al **Operador**.

NOTA – La información técnica sobre dicho traspaso está fuera del alcance de las especificaciones **ECI**.

- Protocolo de transporte de difusión normalizado (que permite una única reproducción en la cabecera de difusión del **Operador**).
- Descubrimiento, implementación del protocolo de transporte y elección de los parámetros del protocolo de transporte operacional normalizados en los receptores.

El flujo de transporte de difusión del **Anfitrión ECI** y las implementaciones del **CPE** serán conformes con DVB SSU [ETSI TS 102 006], y consiguientemente, cumplirán las secciones pertinentes de la definición del carrusel de datos DVB [ETSI EN 301 192], las directrices relativas a la implementación [ETSI TR 101 202] y la definición del carrusel de datos MPEG [ISO/CEI 13818-6].

Los **Operadores** y los **CPE** soportarán el perfil sencillo DVB-SSU y, opcionalmente, el perfil DVB-SSU UNT.

Los **Operadores** pueden soportar varios carruseles simultáneamente.

Los **CPE** explorarán todos los carruseles debidamente señalizados en la SI (información de servicio), la UNT (tabla de notificación de actualización) (si procede) y la PMT (tabla de mapa de programa) para los elementos de descarga pertinentes.

En la Figura 6.4.2.1-1 se muestra el esquema de difusión general para la descarga de imágenes.

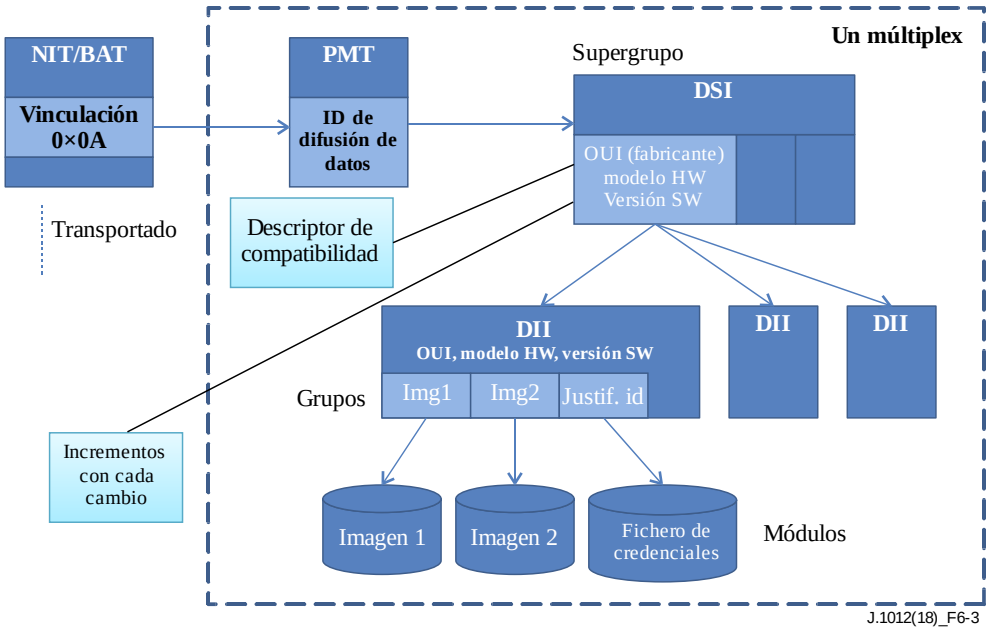


Figura 6.4.2.1-1 – Visión general de la señalización y estructura del carrusel de la Imagen de Anfitrión (variante no basada en la UNT)

6.4.2.2 Traspaso del Fabricante del CPE al Operador

Cualquier futuro ecosistema basado en la **ECI** definirá directrices para que **Operadores** y **Fabricantes de CPE** intercambien de forma uniforme información de ficheros de imágenes (tanto de **Anfitrión ECI** como aquellas que no sean **Imágenes de Anfitrión ECI**), credenciales de imagen **ECI** y metainformación relativa a la descarga desde (numerosos) **Fabricantes de CPE** a (numerosos) **Operadores**.

6.4.2.3 Señalización de SI DVB

6.4.2.3.1 Señalización de la ubicación de la descarga

Los **Operadores** soportarán el descriptor de vinculación DVB-SSU (tipo de vinculación 0x09) como mínimo con el DVI OUI genérico (es decir, vinculación no específica de **Fabricante** con todos los carruseles) en todas las tablas NIT (terrestre o cable) o BAT (satélite).

Los **CPE** de perfil sencillo deberán soportar el descriptor de vinculación DVB-SSU (tipo de vinculación 0x09).

Los **Operadores** que soporten el perfil de la tabla UNT DVB-SSU soportarán también el descriptor de vinculación de exploración SSU (tipo de vinculación 0xA) en todas las tablas NIT (terrestre o por cable) o BAT (por satélite).

Los **CPE** con perfil UNT soportarán el descriptor de vinculación de exploración DVB-SSU (tipo de vinculación 0x09).

6.4.2.3.2 Actualizaciones de emergencia

Para indicar la necesidad de sustituir urgentemente una **Imagen de Anfitrión ECI**, pueden colocarse uno o más descriptores ECI_host_emergency_download en la NIT, la BAT o en alguna de las entradas de la SDT para un servicio al que el **Anfitrión ECI** señalizado pueda proporcionar acceso. El **Anfitrión ECI** también podrá obtener este descriptor de cualquiera de las tablas en las que aparezca en cualquiera de los multiplexores actualmente sintonizados y realizar el procesamiento asociado, así como utilizar cualquier sintonizador de respaldo cuyo estado sea encendido, a fin de acceder periódicamente a los multiplexores pertinentes para adquirir este descriptor a intervalos de

30 minutos como máximo. Se recomienza una mayor frecuencia de verificación de multiplexores no sintonizados (intervalos de 3 minutos).

El ECI_host_emergency_download_descriptor permite poner el objetivo en plataformas de operación y en **Operaciones de Plataforma** e imágenes de cliente específicas a fin de minimizar el número de **Usuarios** que puedan verse afectados negativamente por actualizaciones de emergencia.

Cuando el **Anfitrión ECI** detecte un nuevo descriptor ECI_host_emergency_download comparará la configuración de su **Anfitrión ECI** y de su **Cliente ECI** con la información objetivo del descriptor. Si se detecta una correspondencia entre objetivos y la versión de la imagen de anfitrión instalada necesita una actualización, el **Anfitrión ECI** se actualizará con arreglo al emergency_indicator. Ello interrumpirá las actividades en curso del **Usuario** en el **CPE**.

El descriptor de operaciones **ECI** es un descriptor privado DVB y siempre estará precedido en la tabla donde aparece por el descriptor DVB private_data_specifier_descriptor (véase [ETSI EN 300 468] y [ETSI TS 101 211]) utilizando el private_data_specifier_field de la **ECI**. La sintaxis del descriptor se define en el Cuadro 6.4.2.3.2-1.

Cuadro 6.4.2.3.2-1 – Descriptor ECI_host_emergency_download_descriptor

Sintaxis	N.° de bits	Mnemónico
ECI_host_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* bucle principal */		
main_loop_nr	8	uimsbf
for (i=0; i<main_loop_nr; i++){		
/* bucle de cliente */		
client_nr		
for (j=0; j<client_nr; j++){		
platform_operation_tag	8	uimsbf
Reserved	3	
client_flag	1	
client_tag	4	uimsbf
}		
/* bucle de la imagen de anfitrión */		
host_nr	8	uimsbf
for (j=0; j<host_nr; j++){		
Reserved	4	
emergency_indicator	4	uimsbf
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
min_host_version	8	uimsbf
}		
}		
/* datos privados hasta el final del descriptor*/		
for (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para el descriptor_tag: véase [b-UIT-T J Supl. 7].
descriptor_length	Véase [ETSI EN 300 468].
main_loop_nr	Número de entradas en el bucle principal. El **Anfitrión ECI** evaluará por separado las distintas entradas del bucle principal, es decir, tendrán una semántica OR. Los diversos elementos de una entrada de bucle tendrán una semántica AND.
client_nr	Número de entradas en el bucle objetivo del cliente; el valor 0x00 significa que podrá existir concordancia para cualquier cliente. Las distintas entradas del bucle tendrán una semántica OR y en la actualización de emergencia se considerarán todos los clientes para los que exista concordancia. Los campos de una entrada del bucle tendrán una semántica AND.
platform_operation_tag	El valor de la etiqueta para la **Operación de Plataforma ECI** será la enumerada en el ECI_platform_operation_descriptor de la NIT/BAT. El **Anfitrión ECI** considerará una actualización de emergencia si platform_operation concuerda con platform_operation de alguno de los Clientes instalados.
client_flag	Señaliza si el campo client_tag es pertinente a efectos de concordancia. El valor =0b0 significa que no es pertinente (es decir, cualquier client_id concordará), el valor=0b1 significa que client_tag es pertinente.
host_tag	Valor de la etiqueta que identifica el **Anfitrión ECI** enumerado en el descriptor ECI_platform_operation_descriptor de la NIT/BAT que concuerda con el campo platform_operation_tag en la misma entrada de bucle de cliente. El **Anfitrión ECI** considerará llevar a cabo una actualización de emergencia si vendor_id y client_id se corresponden con alguno de los clientes instalados en el **Anfitrión ECI** para la **Operación de Plataforma.**
host_nr	Número de entradas en el bucle anfitrión. El valor mínimo será 1. Las entradas del bucle tendrán una semántica OR; es decir, si alguna especificación de anfitrión concuerda con la condición objetivo, el estado del bucle principal es de concordancia.
emergency_indicator	El **Anfitrión ECI** utilizará el valor de este campo para seleccionar el comportamiento adecuado para iniciar la descarga y la subsiguiente actualización del anfitrión tal como se define en el Cuadro 6.4.2.3.2-2.
manufacturer_id	Manufacturer_id del anfitrión objetivo en una actualización de emergencia. El **Anfitrión ECI** considerará una actualización de emergencia si el valor de este campo concuerda con el manufacturer_id del **Anfitrión ECI**.
cpe_type_id	Valor definido por ECI_CPE_Type_ID en el Cuadro 6.2.2.1-2. El **Anfitrión ECI** considerará una actualización de emergencia si el cpe_type_id del Anfitrión concuerda con el valor de este campo. El valor 0x000 de cpe_type_id.cpe_type indica que existe concordancia para cualquier cpe_types de **Anfitrión ECI** (ignorándose cpe_model y host-version). Un valor 0x00 de cpe_type_id.cpe_model significa la concordancia de cualquier cpe_model de **Anfitrión ECI** (ignorándose la versión del anfitrión).
min_host_version	El **Anfitrión ECI** considerará una actualización de emergencia si y solo si la versión de su anfitrión es igual o menor que el valor de este campo. NOTA – El valor 0xFF del campo implica la concordancia de todas las versiones de anfitrión.
private_data_byte	Datos privados: el contenido puede ser definido por el **Operador** que gestiona la difusión de este descriptor.

En el Cuadro 6.4.2.3.2-1 se definen varias condiciones del bucle principal (con semántica AND) que deberán cumplirse para que el **Anfitrión ECI** considere una actualización de emergencia. Si se cumplen todas estas condiciones, el **Anfitrión ECI** realizará una descarga de emergencia y la instalación de una nueva imagen de anfitrión de acuerdo con el campo emergency_indicator para ese **Anfitrión ECI**. Los valores del campo indicador se definen en el Cuadro 6.4.2.3.2-2.

Cuadro 6.4.2.3.2-2 – Valores del campo ECI_host_emergency_download_descriptor emergency_indicator

Nombre	Valor	Descripción
System emergency (emergencia del sistema)	0x01	El **Anfitrión ECI** descargará la nueva imagen de anfitrión y la instalará tan pronto como sea posible, interrumpiendo las actividades en curso del **Usuario** si fuera necesario. Véase la Nota.
Regular urgency (urgencia ordinaria)	0x03	El **Anfitrión ECI** descargará la nueva imagen de anfitrión y la instalará en la primera ocasión en la que no tenga que interrumpir actividad alguna del **Usuario.** El **Anfitrión ECI** descargará la nueva imagen de anfitrión a más tardar durante el siguiente evento de activación energética. NOTA – Los **Operadores de Plataforma** pueden utilizarla, por ejemplo, si el **Anfitrión ECI** actual tiene problemas importantes para desencriptar servicios pero puede ejecutar razonablemente casos de uso normales del Usuario.
RFU	otros	Reservado para uso futuro.
NOTA – Los **Operadores de Plataforma** pueden utilizarlo, por ejemplo, si el **Anfitrión ECI** actual tiene problemas importantes de calidad de funcionamiento con las combinaciones objetivo plataforma/cliente.		

6.4.2.4 Señalización PSI

Los **Operadores** soportarán el data_broadcast_id_descriptor de la PMT [ETSI EN 300 468] para cada uno de los carruseles transmitidos, pero no están obligados a soportar cualquier señalización OUI presente en los bytes del selector de este descriptor.

Los **CPE** de perfil sencillo SSU utilizarán el data_broadcast_id_descriptor para ubicar el PID del flujo que transporta un carrusel DVB-SSU.

6.4.2.5 Opción UNT

Esta cláusula sólo se aplica a los **CPE** y **Operadores** que soportan el perfil UNT.

En la PMT se utilizará el data_broadcast_id_descriptor que contenga la estructura system_software_update_info con un valor update_type de 0x2 y el campo OUI puesto a DVB OUI 0x00015A.

Los **Operadores** transportarán una entrada de la tabla SSU en una de las tablas SSU para cada uno de los tipos de **CPE** que soportan.

Los **Anfitriones ECI** deberán poder interpretar los siguientes descriptores UNT (véase [ETSI TS 102 006]):

- SSU_location_descriptor (si se difunde un carrusel para el tipo de **CPE**).
- Scheduling_descriptor (si se prevé disponer en un futuro previsible de un carrusel para el tipo de **CPE**).
- Message_descriptor.

Los **CPE** podrán realizar la descarga coherente y satisfactoria de un carrusel recibido prácticamente sin errores que se configure y desconfigure en los plazos de tiempo nominales publicados y que realice dos ciclos completos (repetición de todos los mensajes del carrusel) en el supuesto de que no exista una actividad de **Usuario** que interfiera con la descarga.

6.4.2.6 Estructura del carrusel

Los carruseles SSU DVB de la **ECI** (para más información véase [ETSI TS 102 006]) utilizarán carruseles de datos de dos capas.

Los carruseles SSU DVB de la **ECI** utilizarán el mensaje DSI con las restricciones siguientes:

- Existirá una lista completa de todos los grupos disponibles para su descarga.
- Cada grupo corresponderá a un **cpe_type + cpe_model** de un **Fabricante**, y dispondrá de todos los recursos para el **Anfitrión ECI** del tipo de **CPE**. Ello implica la disponibilidad de un máximo de 255 módulos (ficheros imagen) (más un fichero con las credenciales).

NOTA 1 – Debido a limitaciones en los valores de **ECI_host_id.model_id** el límite es 239.

- El CompatibilityDescriptor del campo GroupCompatibility de la estructura GroupInfoIndication (para más información véase [ETSI TS 102 006]) aplicará los convenios siguientes:
  - El bucle contendrá un descriptor del hardware del sistema:
    - El OUI corresponderá al **Fabricante** del **CPE**.
    - Los campos modelo y versión asociados con el descriptor del hardware del sistema corresponderán a los **cpe_type** y **cpe_model** del CPE y serán iguales a los campos **id.cpe_type** e **id.cpe_model** del **Certificado de Anfitrión ECI** del fichero de credenciales del grupo.
  - El bucle incluirá un descriptor del software del sistema; el campo modelo se pondrá a 0, el campo versión reflejará la versión del software del **Anfitrión ECI** en el grupo (es decir, tanto **Imágenes de Anfitrión ECI** como **Imágenes** que no correspondan al **Anfitrión ECI**).

Los **CPE** utilizarán los campos modelo y versión de compatibilityDescriptor para compararlos con su propio modelo y versión de **CPE** y utilizará el campo versión del software para verificar si el grupo contiene una actualización y en caso de que se trate de una nueva versión proceder a descargar nuevas imágenes.

El carrusel SSU DVB de la **ECI** utilizará los campos del mensaje DII con las restricciones siguientes:

- El blockSize tomará un valor de 2 kbyte (2 048 byte) como mínimo.
- El valor asignado al campo "tDownloadScenario" reflejará un tiempo de descarga de todos los módulos de como mínimo a 4 veces el tiempo de repetición más lento del mensaje (tiempo de compleción del carrusel).
- Los bits 7…0 del moduleId bits coincidirán con el **id.image_model** del fichero imagen.
- La moduleVersion será igual al **id.image_version ECI** del fichero imagen.

Los **CPE** pueden utilizar el campo "tDownloadScenario" para terminar descargas que no fueron exitosas (por ejemplo, debido a elevadas tasas de errores en los paquetes) e informar del problema al **Usuario**.

El grupo de un tipo de **CPE** contendrá los módulos siguientes:

- Ficheros imagen para un tipo de **CPE** (puede ser un conjunto de imágenes parcial).
- El fichero de credenciales de la **Imagen de Anfitrión ECI** que contenga las (últimas) credenciales de todas las imágenes de un **Anfitrión ECI**:
  - los bits 7..0 del moduleId del DII de este módulo tendrán el valor 0xFF; y
  - moduleVersion se incrementará con cada cambio.

NOTA 2 – Los **Operadores** pueden compartir ficheros comunes entre descargas para varios tipos de **CPE** compartiendo DownloadDataBlocks entre los DII. No obstante, ello implica gestionar de forma consistente los Id de **Imagen de Anfitrión ECI** entre los tipos de **CPE**.

6.4.2.7 Operación de descarga del Anfitrión ECI

El cargador de la **Imagen de anfitrión ECI** intentará verificar cada 30 minutos que todos los carruseles posibles que se encuentran en situación de alimentación de energía activa tienen disponibles los recursos de red y al menos cada 6 horas para aquellos que se encuentren en estado energético de espera, sin perturbar al **Usuario**, por ejemplo, una vez que el **CPE** pasa al estado de espera y durante periodos que no coincidan con la máxima audiencia.

Si un proveedor de red pone a disposición tablas UNT que contengan posibles descargas para un tipo de **CPE**, los correspondientes **CPE** verificarán regularmente las UNT para programar una posible nueva actualización. El **CPE** intentará verificarlas con la misma frecuencia que en el caso de los carruseles de imagen de **Anfitrión ECI**.

Se recomienda que el **Usuario** reciba un aviso si un **CPE** en modo sólo difusión no puede realizar las verificaciones indicadas durante más de 2 semanas.

Una vez detectada la disponibilidad de una nueva descarga, lo que significa que el **CPE** y el **Usuario** han dado su conformidad, el **CPE** tratará de realizar la descarga e instalar la nueva imagen (posiblemente sobrescribiendo sobre una versión anterior). El **Usuario** será informado de cualquier fallo continuado en la realización de una descarga. Los **Anfitriones ECI** siempre podrán subsanar un error de descarga de una imagen recuperando un determinado estado funcional, por ejemplo, restaurando la imagen de anfitrión previa o volviendo a intentar la descarga de la nueva imagen de anfitrión.

Debe señalarse que el fallo continuado en la descarga de nuevos ficheros **Imagen de Anfitrión ECI** o de credenciales puede causar que el **Operador** deniegue el servicio.

6.4.2.8 Programación asociada a los carruseles del Operador

Los **Operadores** deben proporcionar suficiente anchura de banda para que la descarga de carruseles de datos de imágenes del **CPE** se realice en un tiempo razonable.

6.4.2.9 Aspectos relativos a la interfaz de Usuario

Un **CPE** que pueda realizar descargas de ficheros **Imagen de Anfitrión ECI** en una red de difusión deberá:

- disponer de un modo de exploración de descarga que automatice con regularidad la verificación de disponibilidad de nuevas imágenes o credenciales, por ejemplo, formando parte de un estado de espera, y con una configuración que se recomienda que sea la configuración por defecto del **Fabricante** para verificar las descargas; y
- disponer de un ajuste en el menú del CPE que automatice cualquier aprobación por el **Usuario** de la aceptación de nuevos ficheros **Imagen de Anfitrión ECI** o credenciales, que se recomienda que sea la configuración por defecto del **Fabricante** para aprobar las descargas.

Los **CPE** dispondrán, al menos, de una forma alternativa de descarga de nuevos ficheros **Imagen de Anfitrión ECI** a fin de evitar que los **CPE** que operen en redes de difusión que no proporcionen nuevos ficheros **Imagen de Anfitrión ECI** para sus tipos de CPE sufran denegación del servicio.

6.4.3 Protocolo de transporte de Internet del Anfitrión ECI

6.4.3.1 Protocolo IP

La **ECI** no define un protocolo específico para que un **CPE** verifique la disponibilidad de nuevos ficheros **Imagen de Anfitrión ECI** de un proveedor de servicio proporcionados por el **Fabricante**. No obstante, se recomienda utilizar HTTP1.1 [IETF RFC 7231] como protocolo de transferencia de ficheros y el protocolo definido en la cláusula 7.7.3.3, que define un servicio normalizado de descarga de ficheros **Imagen de Anfitrión ECI** desde un servidor de **Operación de Plataforma**.

Normalmente, el **Fabricante del CPE** proporciona el servidor de descarga de la **Imagen de Anfitrión ECI**. En virtud de acuerdos específicos entre el **Fabricante del CPE** y un **Operador** (o de terceros que actúen en su nombre), también puede proporcionarlo el **Operador** o un tercero.

6.4.3.2 Funcionamiento del cargador en línea

El cargador de **Imagen de Anfitrión ECI** en línea de la **ECI** intentará verificar el estado de su servidor en línea cada 30 minutos sin perturbar al **Usuario**. Se recomienda avisar al **Usuario** si un **CPE** que únicamente tiene el modo de funcionamiento en línea no puede realizar las verificaciones anteriores durante un periodo más prolongado.

Una vez que se detecta la disponibilidad de una nueva descarga, el **CPE** tratará de realizarla e instalar la nueva imagen (posiblemente sobrescribiendo sobre versiones anteriores de la imagen). Se informará adecuadamente al **Usuario** de cualquier fallo continuado en la descarga.

Debe señalarse que el fallo en la descarga de nuevas **Imágenes de Anfitrión ECI** o de credenciales puede causar que el **Operador** deniegue el servicio.

El cargador en línea del **CPE** distribuirá un conjunto de (nuevas) imágenes y de credenciales de imágenes tal como se define en la cláusula 6.3 para su verificación, almacenamiento y activación.

El cargador en línea de **Imágenes de Anfitrión ECI** tendrá características de descarga de emergencia con el mismo efecto que el definido en la cláusula 6.4.2.3.2 para la difusión.

6.4.4 Protocolos de transporte alternativos

Un **Anfitrión ECI** puede usar cualquier protocolo de distribución alternativo (propiedad del fabricante).

El cargador del **CPE** procesará un conjunto de (nuevas) imágenes y de credenciales de imágenes, tal como se define en la cláusula 6.3 para su verificación, almacenamiento y activación.

7 Cargador de Cliente ECI

7.1 Introducción

El **Anfitrión ECI** puede descargar, almacenar y activar **Imágenes de Cliente ECI** y datos conexos. El proceso de carga del **Cliente ECI** puede subdividirse en las fases siguientes:

1) Descubrimiento de la protección basada en la **ECI** de un servicio/paquete de servicios y/o otras formas de identificar la necesidad de un **Cliente ECI**. Forma parte de la aplicación de navegación ordinaria del **CPE**.
2) Determinación de la ubicación en red (difusión o en línea) de los recursos necesarios para instalar el **Cliente ECI** en el **Anfitrión ECI**.
3) Descarga y almacenamiento (en memoria NV) de información de la **Operación de Plataforma** necesaria para instalar el Cliente ECI y verificar las credenciales.
4) Registro del **Anfitrión ECI** en el sistema de seguridad de la **Operación de Plataforma** y recepción (si es necesario) de datos de inicialización específicos del CPE para la desencriptación del **Cliente ECI**.
5) Descarga desde la red y almacenamiento (en memoria NV) de la **Imagen de Cliente ECI** y de las credenciales del **Cliente ECI** asociadas, verificación de las credenciales de la imagen, y su almacenamiento en memoria NV para uso futuro.
6) Inicialización del **Cliente ECI** utilizando la **Imagen de Cliente ECI**, el **Certificado de Operación de Plataforma**, la atribución de un Contendor **ECI** y los recursos de AS necesarios e inicio de la ejecución del **Cliente ECI**.

Todos los procesos pueden ejecutarse utilizando datos del flujo de difusión o de Internet, con excepción de registro del **CPE** en el **Operador**, que requiere asistencia manual si sólo hay disponible una conexión de difusión.

Los **Operadores** pueden renovar los recursos de **Cliente ECI** en cualquier momento mediante la publicación de información en las redes de difusión o en línea. El **Anfitrión ECI** verifica regularmente dichas actualizaciones.

La **ECI** necesita datos de apoyo para diversas funciones de un **CPE**, como por ejemplo, para los datos de revocación o las **Cadenas de Certificación** actualizadas que necesita el **Cliente ECI** y/o el **Anfitrión ECI** para soportar al **Cliente ECI**. En las redes de difusión el protocolo de transporte permite descargar selectivamente los datos que necesita un **CPE** sobre la base de un índice ("hash") de la identificación de los datos. La agrupación de datos según el hash del índice se denomina "colectación" ("*bucketizing*"). La descarga selectiva en redes en línea se basa en transferir la identificación de los datos necesarios como parámetros a una API de servicios en Internet.

El **Anfitrión ECI** puede descargar los elementos de datos siguientes:

- **Imágenes de Clientes ECI** (en formato de colector en redes de difusión).
- Datos de revocación del **Cliente ECI** (en formato colector en redes de difusión).
- Cadena de cliente Operación de Plataforma.
- Datos de revocación de **Operación de Plataforma** (en formato colector en redes de difusión).
- Datos de revocación de la **Imagen de Cliente ECI** (en formato colector en redes de difusión). Datos de inicialización de cliente en configuración de AS ECI para la desencriptación de imágenes de cliente encriptadas (en formato colector en redes de difusión).

7.2 Descubrimiento de Clientes ECI

7.2.1 Introducción

Normalmente, un **CPE** conforme con la **ECI** (por ejemplo, un iDTV) no tendrá instalado ningún **Cliente ECI** cuando sale de fábrica, ya que el dispositivo puede venderse en cualquier mercado en el mundo. En la cláusula siguiente se define el mecanismo que permite a un **CPE** conforme con la **ECI** localizar **Clientes ECI** para la descodificación de servicios distribuidos en la red a la que está conectado.

En relación con el proceso de descubrimiento se distinguen dos tipos de redes:

1) Redes basadas en flujos de transporte (redes de difusión y redes IPTV típicas).

2) Redes basadas en el protocolo IP.

La **ECI** soporta dos modos de descubrimiento de proveedor y de cliente para redes basadas en flujos de transporte:

1) Instalación manual – incluidos los parámetros de configuración de red básicos (difusión).

2) Autodescubrimiento (con elección del **Usuario**) – asume que el **CPE** puede realizar la autoinstalación para la red en cuestión de forma autónoma.

Los protocolos de instalación manual y de autodescubrimiento en redes de flujos de transporte utilizan señalización común.

En el caso de redes basadas en el protocolo IP, la **ECI** permite la introducción manual de los URL.

7.2.2 Redes basadas en flujos de transporte

7.2.2.1 Señalización común

Para reducir la introducción manual de parámetros por el **Usuario**, la **ECI** proporciona señalización en línea de parámetros **ECI** que son clave para la instalación de un cliente:

- Uno o más ECI_platform_operation_descriptors en la NIT que transporta los clientes disponibles (según el ID) para cada **Platform_Operation**. El descriptor incluye el nombre del proveedor de la plataforma y un id abreviado (para permitir la representación compacta en la cadena de instalación manual).
- Un proveedor de plataforma puede especificar un URL base para la API web en el ECI_base_URL_descriptor.

7.2.2.2 Descriptor de operación de plataforma ECI (ECI_platform_operation_descriptor)

El ECI_platform_operation_descriptor proporciona información clave sobre una **Operación de Plataforma** que ofrece servicios de acceso para una red de flujos de transporte.

Para cada **Platform_Operation** la NIT_{real} (y/o BAT en redes por satélite) transportará como mínimo el ECI_ platform_operation_descriptor en el múltiplex central y la tabla identificados en la cadena de instalación de redes que sólo disponen de instalación manual y en todos los multiplexores salvo en redes por satélite y redes con autodescubrimiento. Las redes por satélite solo pueden transportar el ECI_platform_operation_descriptor en los multiplexores en los que el proveedor transporta servicios: como parte de la NIT o de una BAT.

El ECI_platform_operation_descriptor es un descriptor privado DVB que utiliza el especificador de datos privados de la **ECI** en el private_data_specifier_descriptor DVB [ETSI TS 101 162]. Se define en el Cuadro 7.2.2.2-1.

Cuadro 7.2.2.2-1 – ECI_platform_operation_descriptor

Sintaxis	N.° de bits	Mnemónico
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
/* bucle del nombre de la plataforma */		
for (i=0; i<N; i++){		
platform_name_char	8	uimsbf
}		
for (i=0; i<N; i++){		
extension_byte	8	uimsbf
}		
}		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para descriptor_tag, véase [b-UIT-T J Supl. 7].
platform _tag	Este campo de 8 bits especifica la etiqueta de la **Platform_Operation** para instalación manual. Los NIT y BAT de la red que soporte cada **Platform_Operation** tendrán un valor único de platform_tag. Cada platform_tag aparecerá solo una vez en cada NIT o BAT. La platform_tag no se utilizará para la ordenación de los proveedores y no estará presente en la interfaz de **Usuario CPE** para la selección de la **Platform_Operation**.
operator_id	ID de Operador definido en la cláusula 7.5.2 de la presente Recomendación. Es el identificador del **Operador** de la **Platform_Operation**.
platform_operation_id	ID de **Platform_Operation** tal como se define en 7.5.3 de la presente Recomendación.
platform_name_length	Longitud de la secuencia de octetos del bucle nombre de la plataforma. Si la longitud es 0, el proveedor no permitirá el autodescubrimiento y no se incluirá en ningún menú de selección de proveedor del menú de instalación de cliente del **CPE**. El valor máximo de este campo será 40.
platform_name_char	Secuencia de caracteres UTF8 que representa el nombre de la operación de plataforma.
extension_byte	Bytes adicionales; reservados para uso futuro por esta Recomendación.

7.2.2.3 Descriptor de url base de ECI (ECI_base_url_descriptor)

El ECI_base_url_descriptor permite a la **Platform_Operation** señalizar el URL base de su API web (véase la cláusula 7.7.3) que puede utilizarse para proveer servicios conexos a la instalación de cliente en caso de acceso en línea.

Para cada **Platform_Operation** la NIT_{real} (y/o BAT en redes por satélite) puede transportar el ECI_base_url_descriptor en la misma tabla que transporte el ECI_platform_operation_descriptor.

El ECI_base_url_descriptor es un descriptor privado DVB que utiliza el especificador de datos privados de la **ECI** en el private_data_specifier_descriptor DVB [ETSI EN 300 468]. Se define en el Cuadro 7.2.2.3-1.

Cuadro 7.2.2.3-1 – ECI_base_url_descriptor

Sintaxis	N.° de bits	Mnemónico
`ECI_base_url_descriptor(){`		
`  descriptor_tag`	8	uimsbf
`  descriptor_length`	8	uimsbf
`  platform_tag`	4	uimsbf
`  reserved`	4	
`  base_url_length`	8	uimsbf
`  /* bucle de url base */`		
`  for (i=0; i<N; i++){`		
`  base_url_char`	8	uimsbf
`  }`		
`}`		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para descriptor_tag, véase [b-UIT-T J Supl. 7].
platform_tag	Este campo de 4 bits especifica la etiqueta del proveedor para la instalación manual. Cada NIT y BAT de la red que soporte a cada **Platform_Operation** tendrá un valor único de platform_tag. Cada platform_tag aparecerá solo una vez en cada NIT o BAT. La platform_tag no se utilizará para la ordenación de **Platform_Operations** y no estará presente en la interfaz de **Usuario CPE** para la selección de la **Platform_Operation**.
base_url_length	Este campo indicará el número de octetos en el bule URL base.
base_url_char	Secuencia de caracteres UTF8 que conforma el URL base para una operación de plataforma.

7.2.2.4 Instalación manual

La **Platform_Operation** puede proporcionar una cadena de instalación al **Usuario** que puede introducirla en un elemento adecuado del menú de instalación de la interfaz de **Usuario CPE** a fin de instalar un **Cliente CPE**. La cadena de instalación se definirá de conformidad con esta cláusula. La cadena de instalación es una representación numérica binaria de longitudes variables. El número binario, representado con el bit más significativo en primer lugar, puede construirse concatenando los valores binarios de 3 bits de los dígitos donde el bit más significativo figura en primer lugar.

El número se presenta al **Usuario** en fragmentos de 4 dígitos y la entrada en la UI del **CPE** serán asimismo fragmentos de 4 dígitos.

La cadena de instalación identifica los parámetros definidos en el Cuadro 7.2.2.4-1.

Cuadro 7.2.2.4-1 – Parámetros de la cadena de instalación (en número de bits)

Parámetro	DVB-T/DVB-T2	DVB-C/DVB-C2	DVB-S/DVB-S2	IPTV	Mnemónico
Tipo de red	3	3	3	3	uimsbf
ID de red	16	17	17	16	uimsbf
Etiqueta de plataforma	8	8	8	8	uimsbf
Etiqueta de cliente	4	4	4	4	uimsbf
Relleno	0	0	0	0	uimsbf
Verificación de suma	5	5	5	5	uimsbf
Número de bits	36	36	36	36	uimsbf
Número de dígitos	12	12	12	12	uimsbf
Número de fragmentos	3	3	3	3	uimsbf

Semántica:

Tipo de red	Campo de 3 bits. En el Cuadro 7.2.2.4-2 se presentan los valores según el tipo de red.
ID de red	Id de la tabla SI DVB que contiene el ECI_service_provider_descriptor (véase 7.2.2.2) que proporciona información adicional necesaria para el acceso a los servicios tal como se define en el Cuadro 7.2.2.4-3.
Etiqueta de plataforma	Campo de 4 bits que representa la etiqueta de proveedor del proveedor de servicio necesario en el ECI_service_provider_descriptor en la NIT o la BAT.
Etiqueta de cliente	Campo de 4 bits que representa la etiqueta de proveedor del cliente necesario en el ECI_service_provider_descriptor seleccionado por la etiqueta de proveedor en la NIT o la BAT.
Relleno	Campo de 0..2 bits con valores a 0 que rellenan la cadena precedente hasta alcanzar un múltiplo de 3 bits.
Verificación de suma	Campo de 5 bits formado por la adición de fragmentos consecutivos de 5 bits de la cadena precedente. La última parte de la cadena se rellena en su inicio con ceros hasta una longitud de 5 bits. Por ejemplo, la verificación de suma de la cadena 0b01011010 es 0b01011 + 0x00010 = 0b01101. En la interfaz de **Usuario** del **CPE** se utilizará la verificación de suma para rechazar cualquier entrada errónea por parte del **Usuario**.

Cuadro 7.2.2.4-2 – Representación del valor del tipo de red

Tipo de red	Valor
DVB-T/T2	0
DVB-C/C2	1
DVB-S/S2	2
IPTV	3
Reservado	4..7

Cuadro 7.2.2.4-3 – Representación del ID de red

Tipo de red	Valor del ID de red	Número de bits
DVB-C	0b0 seguido del ID de red de la tabla NIT o 0b1 seguido del ID de BAT de la tabla BAT	17
DVB-S/S2	0b0 seguido del ID de red de la tabla NIT o 0b1 seguido del ID de BAT de la tabla BAT	17 17

7.2.2.5 Instalación mediante autodescubrimiento

En este método de instalación el **CPE** podrá autodescubrir los parámetros de red de la red de flujos de transporte y, por lo tanto, acceder a todos los flujos de transporte de esa red.

Cada servicio en cada uno de los multiplexores se etiquetará con la etiqueta **Platform_Operations ECI** que puede proporcionar acceso al servicio. Esto puede hacerse en la SDT para cada servicio (véase la cláusula 7.2.2.6) o en la NIT o la BAT (sólo para redes por satélite) para cada múltiplex (véase la cláusula 7.2.2.6).

El **CPE** ofrecerá al **Usuario** la opción de instalar cualquier **Cliente ECI** de la **Platform_Operations** como parte del proceso de instalación mediante autodescubrimiento. En caso de que un **Usuario** decida instalar un **Cliente ECI** de la **Platform_Operations** para recibir servicios desencriptados a través de la red de acceso conexa, el comportamiento por defecto del **CPE** será instalar todos los **Servicios** etiquetados asociados a esa **Operación de Plataforma** en la lista de servicios principal del **CPE**.

7.2.2.6 Descriptor de la etiqueta de servicio ECI

El ECI_service_tag_descriptor se transporta en la SDT. Etiqueta cada servicio con los proveedores de servicio **ECI** con capacidad para desaleatorizar el servicio. La definición se muestra en el Cuadro 7.2.2.6-1.

Cuadro 7.2.2.6-1 – Descriptor de la etiqueta de servicio ECI

Sintaxis	N.° de bits	Mnemónico
`ECI_service_tag_descriptor(){`		
`descriptor_tag`	8	uimsbf
`descriptor_length`	8	uimsbf
`platform_tag`	8	uimsbf
`}`		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para el descriptor_tag, véase [b-UIT-T J Supl. 7].
platform_tag	Valor de platform_tag de la **Platform_Operation** de la **ECI**, tal como se enumera en el ECI_platform_operation_descriptor, transportado en la NIT o en la BAT de la red.

7.2.2.7 Descriptor de la lista de la plataforma ECI

El descriptor de la lista de la plataforma ECI proporciona la lista de **Platform_Operations** de la **ECI** que da acceso a los servicios de los distintos multiplexores de la red. El ECI_platform_list_descriptor se transporta en la NIT y/o la BAT. La definición se muestra en el Cuadro 7.2.2.7-1.

Cuadro 7.2.2.7-1 – ECI_platform_list_descriptor

Sintaxis	N.° de bits	Mnemónico
ECI_platform_list_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
for (i=0;i<N;i++){		
platform_count	8	uimsbf
/* bucle de plataforma */		
for (j=1; j<M; j++){		
platform_tag	8	uimsbf
}		
service_count	16	uimsbf
/* bucle del servicio */		
for (j=0; j<M; j++){		
service_id	16	uimsbf
}		
}		
}		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para el descriptor_tag, véase [b-UIT-T J Supl. 7].
platform_count	Campo de 8 bits que es el número de etiquetas de proveedor en el bucle siguiente.
platform_tag	Valor de platform_tag de la Platform_Operation ECI, tal como se enumera en el ECI_platform_operation_descriptor, transportado en la NIT o en la BAT de la red. En el bucle de servicio se indican los servicios a los que está asociada la Platform_Operation etiquetada. Los valores de Platform_tag pueden aparecer varias veces en el bucle externo de este descriptor.
service_count	Campo de 16 bits, que representa el número de los identificadores de servicio (service_id) incluidos en el bucle siguiente.
service_id	ID de servicio DVB de un servicio en el multiplex de la NIT o la BAT al que se accede utilizando los servicios de acceso de las plataformas a las que se hace referencia en el bucle de plataforma precedente.

7.2.3 Descubrimiento de cliente en redes IP

7.2.3.1 Instalación manual

Un **CPE** con acceso a redes IP ofrecerá la opción de entrada manual del URL para permitir la instalación de un proveedor de servicio. El URL servirá de como URL base para la API web.

NOTA – El **CPE** puede ofrecer acceso a diversos servicios en línea como parte de las funciones de las aplicaciones de un **CPE**, algunas de las cuales pueden ser descargadas. Al objeto de automatizar el proceso de instalación de clientes por el **Usuario**, el **CPE** puede ofrecer un proveedor de servicio junto con una interfaz de API de instalación de cliente.

7.2.3.2 Instalación basada en una página en Internet

Este tipo de solución para la instalación de un **Cliente ECI** está fuera del alcance de la presente Recomendación y puede estar sujeta a especificaciones complementarias.

7.3 Almacenamiento, verificación y activación

7.3.1 Políticas de actualización generales

La **ECI** permite la renovación frecuente de elementos para lograr de una elevada integridad. Por lo tanto, todos los elementos descargados se verifican con frecuencia para mantenerlos actualizados. La siguiente política de descarga de actualizaciones se aplicará a todos los datos de **Clientes ECI** y de **Operaciones de Plataforma**, así como a los datos de revocación asociados.

Los **Anfitriones ECI** tratarán regularmente de identificar actualizaciones e informarán al **Usuario** en caso de que sea necesario adoptar alguna medida. Los requisitos de la política de actualización figuran en [b-UIT-T J Supl. 7].

El **Anfitrión ECI** almacenará la **Cadena del Cliente Operación de Plataforma** con el **Cliente ECI** asociado. El almacenamiento y supresión se gestionará como parte de la instalación y supresión de **Clientes ECI**.

El **Anfitrión ECI** actualizará automáticamente el **Certificado** del proveedor de la plataforma que sobrescribirá versiones anteriores.

7.3.2 Descarga y almacenamiento de la Imagen de Cliente ECI

Como parte de la gestión de recursos conexos del **Cliente ECI**, el **Anfitrión ECI** almacenará la **Imagen de Cliente ECI** necesaria para acceder a servicios o al contenido de la memoria NV sólo tras la aprobación (implícita) por el **Usuario**. Cualquier política automática de instalación de **Clientes ECI** proporcionará un método transparente para el **Usuario** para enfrentar cualquier limitación de recursos a fin de gestionar los **Clientes ECI** de manera transparente al **Usuario** sin que ello requiera la pérdida imprevista del acceso al contenido o a los servicios. En consonancia, cualquier supresión de una **Imagen de Cliente ECI** deberá ser aprobada (implícitamente) por el **Usuario**.

El **Anfitrión ECI** almacenará los **Clientes ECI** descargados en una memoria NV con sus credenciales originales conforme a la **Operación de Plataforma**. Las nuevas versiones de **Cliente ECI** (incluyendo sólo nuevas credenciales) sobrescribirán versiones más antiguas (conforme a la **Operación de Plataforma**). Por ejemplo, si dos **Operaciones de Plataforma** utilizan el mismo tipo de **Cliente ECI** pero distintas versiones del mismo, el **Anfitrión ECI** almacenará ambas versiones.

El mínimo tamaño de imagen que puede almacenar un **CPE** por cada intervalo de **Cliente ECI** se define en [b-UIT-T J Supl. 7].

7.3.3 Validación y activación de Clientes ECI

El **Anfitrión ECI** cargará la última **Cadena de Clientes Operación de Plataforma** (según su número de versión) para el **Certificado de Operación de Plataforma** en el **Sistema de Seguridad Avanzada** e intentará instalar la clave pública de la **Operación de Plataforma** de conformidad con las normas genéricas de procesamiento de cadenas definidas en la cláusula 5.4.2.

El **Anfitrión ECI** cargará el último **Cliente ECI** en el **Sistema de Seguridad Avanzada**. Cargará la cofirma de cliente **Operación de Plataforma** en el **Sistema de Seguridad Avanzada**. A continuación validará el **Cliente ECI** con arreglo a las normas genéricas de procesamiento de cadenas descritas en la cláusula 5.5 y verificará la firma y la cofirma de la **Imagen de Cliente ECI**. El **Anfitrión ECI** lo notificará al **Usuario** cualquier revocación que se produzca.

Sólo se instalará y activará un nuevo **Cliente ECI** si el proceso de validación se completa satisfactoriamente.

7.4 Formatos de la estructura de la Cadena de Cliente ECI

7.4.1 Introducción a los formatos de la estructura de la Cadena de Cliente ECI

En la Figura 7.4.1-1 se presenta un esquema de la estructura de la **Cadena de Certificados de Cliente ECI**. La cadena comienza con la **Lista de Revocación** del Suministrador, seguida del **Certificado del Suministrador de Seguridad**, la **Lista de Revocación del Cliente ECI** y finalmente, el fichero de la **Imagen de Cliente ECI**. En caso de **Series de Imágenes**, se añade un **Certificado de Imagen de Cliente ECI** adicional. La firma del Cliente **Operación de Plataforma ECI** proporciona una segunda firma a la imagen de cliente garantizando la aplicabilidad del **Cliente ECI** a la operación de plataforma. Se define en la cláusula 7.5.

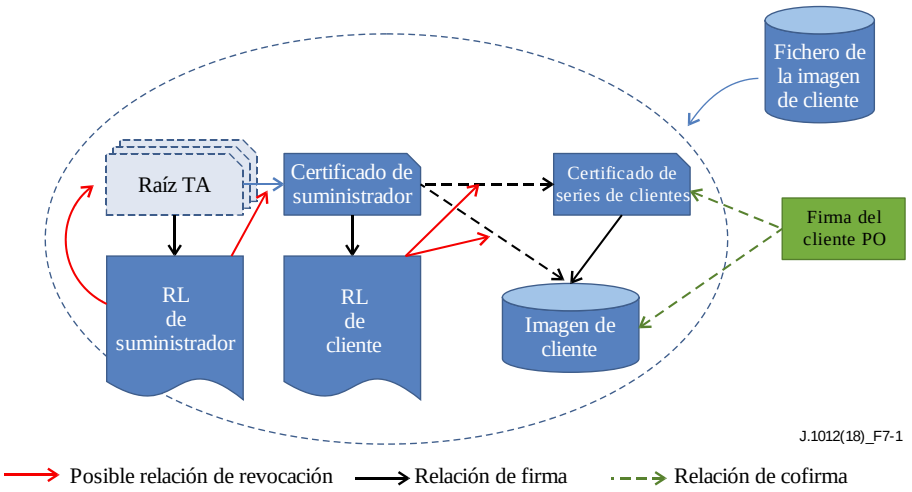


Figura 7.4.1-1 – Cadena de autenticación del cliente

7.4.2 Certificado de Suministrador de Seguridad

Los **Certificados de Suministrador de Seguridad** se definen en la estructura de ECI_Certificate. El ID de **Certificado** para el **Certificado de Suministrador de Seguridad** se define en el Cuadro 7.4.2-1.

Cuadro 7.4.2-1 – Definición del ID del Suministrador de Seguridad

Sintaxis	N.° de bits	Mnemónico
`ECI_Vendor_Id {`		
`padding(4)`		
`type` `/* véase el Cuadro 5.2-2 */`	4	uimsbf
`vendor_id`	20	uimsbf
`vendor_version`	8	uimsbf
`}`		

Semántica:

type: entero	Valor de conformidad con el Cuadro 5.2-2.
vendor_id: entero	Número de suministrador asignado al Suministrador de Seguridad, único en el contexto de la **ECI**.
vendor_version: entero	Id asignado de forma incremental a la versión del **Certificado** del Suministrador de Seguridad. Los valores 0x00 y 0xF0..0xFF están reservados.

7.4.3 Certificados de series de Clientes ECI e identificador de objetivo de series

La estructura de **ECI_Certificate** define los **Certificados** de series de **Clientes ECI**. El ID de Certificado del **Certificado de Suministrador de Seguridad** se define en el Cuadro 7.4.3-1.

Cuadro 7.4.3-1 – Definición del ID de Series de Clientes

Sintaxis	N.° de bits	Mnemónico
`ECI_Client_Series_Id {`		
`padding(4)`		
`type` `/* véase el Cuadro 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Semántica:

type: entero	Valor de conformidad con el Cuadro 5.2-2.
client_type: entero	Tipo de **Cliente ECI** singular en el contexto del id del Suministrador de Seguridad del **Cliente ECI**.
client_version_major: entero	Número de versión superior ("major") del **Cliente ECI** de un **tipo de Cliente ECI**. La versión se incrementa con cada nueva edición superior (véase la nota).
client_version_minor: entero	Número de versión inferior ("minor") del **Cliente ECI.** Los **Clientes ECI** pueden ser revocados por comparación de un número de versión inferior de las **Listas de revocación de Cliente ECI,** y ser sustituidos automáticamente.
NOTA – La sustitución en un **Cliente ECI** por un cambio de versión superior no es automático en los **CPE** conformes con la **ECI,** ya que sólo se realizan automáticamente las actualizaciones de versiones inferiores.	

NOTA – Los **Certificados** de series de tipo de **Cliente ECI** se asignan a **Clientes ECI** que necesitan implementaciones personalizadas en cada **CPE,** que son idénticas desde una perspectiva de seguridad y funcionalidad.

El ID del objetivo de cliente se define de la misma forma que para los **Anfitriones ECI**, utilizando la estructura de ECI_Host_Series_Image_Target_Id. Ello vincula una imagen de cliente a un **Anfitrión ECI** específico.

7.4.4 Firma de Imagen de Cliente ECI

Las firmas de **Cliente ECI** utilizarán la estructura de ECI_Data_Signature definida en la cláusula 5.6.

El ID de **Cliente ECI** se define en el Cuadro 7.4.4-1, y su estructura es idéntica a la del ECI_Client_Series_Id definido en el Cuadro 7.4.3-1.

Cuadro 7.4.4-1 – Definición de ID de Cliente

Sintaxis	N.° de bits	Mnemónico
`ECI_Client_Id {`		
`padding(4)`		
`type` `/* véase el Cuadro 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Semántica:

type: entero	Valor de conformidad con el Cuadro 5.2-2.
client_type: entero	Tipo de cliente, tal como es asignado por la **TA ECI**.
client_version_major: entero	Número de versión superior ("major") del **Cliente ECI** de un **tipo de Cliente ECI**. La versión se incrementa con cada nueva edición superior.
client_version_minor: entero	Número de versión menor ("minor") del **Cliente ECI**. Los **Clientes ECI** pueden ser revocados por comparación con el número de versión menor de las Listas de revocación de **Cliente ECI**.

7.5 Formatos de la Cadena de Operación de Plataforma ECI

7.5.1 Generalidades

En la Figura 7.5.1-1 se presenta un esquema de la cadena de autenticación para el **Certificado de Operación de Plataforma** y las firmas de cliente **Operación de Plataforma**. Comienza con la **Lista de Revocación de Operador**, seguida del **Certificado de Operador**, la **Lista de Revocación de Operación de Plataforma** y finalmente el **Certificado de Operación de Plataforma** que contiene la clave pública de la **Operación de Plataforma**. Esta se utiliza conjuntamente con la **Lista de Revocación** del Cliente **Operación de Plataforma** para validar las **Imágenes de Cliente ECI** cuyo funcionamiento se permite en la plataforma.

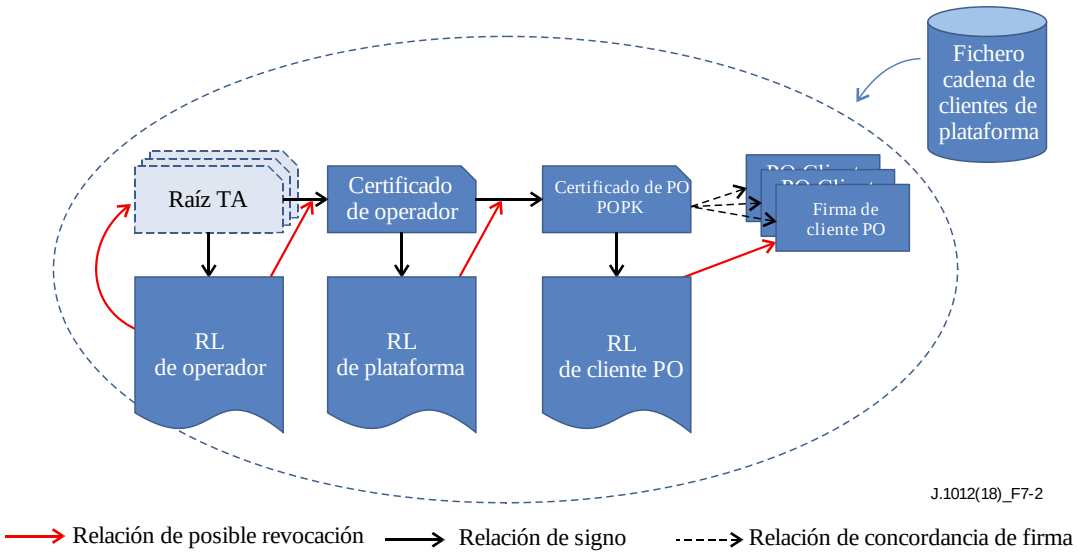


Figura 7.5.1-1 – Cadena de autenticación para la cadena de clientes de plataforma

7.5.2 Certificado de Operador

Los **Certificados de Operador** vienen definidos por la estructura de **ECI_Certificate**. Los identificadores del **Operador** se definen en el Cuadro 7.5.2-1.

Cuadro 7.5.2-1 – Definición del ID del Operador

Sintaxis	N.° de bits	Mnemónico
`ECI_Operator_Id {`		
`padding(4)`		
`type` `/* véase el Cuadro 5.2-2`	4	uimsbf
`operator_id`	20	uimsbf
`operator_version`	8	uimsbf
`}`		

Semántica:

type: byte	Valor de conformidad con el Cuadro 5.2-2.
operator_id: entero	**ID de Operador** asignado a un **Operador**, único en el contexto de la raíz **ECI**.
operator_version: entero	Número de versión asignada incrementalmente a la versión del **Certificado** del **Operador**. Los valores 0x00 y 0xF0..0xFF están reservados.

7.5.3 Certificado de Operación de Plataforma

Los **Certificados de Operación de Plataforma** se definen en la estructura de **ECI_Certificate**. La **Operación de Plataforma** gestiona su clave secreta. El ID de certificado del **Certificado de Operación de Plataforma** se define en el Cuadro 7.5.3-1.

Cuadro 7.5.3-1 – Definición del ID de Operación de Plataforma

Sintaxis	N.° de bits	Mnemónico
`ECI_Platform_Operation_Id {`		
`padding(4)`		
`type` `/* véase el Cuadro 5.2-2`	4	uimsbf
`platform_operation_id`	20	uimsbf
`platform_operation_version`	8	uimsbf
`}`		

Semántica:

type: byte	Valor de conformidad con el Cuadro 5.2-2.
platform_operation_id: entero	Número de **Operación de Plataforma** asignado al Suministrador de Seguridad, único en el contexto del **Certificado** de Operador**.**
platform_operation_version: entero	Aumenta si la **Operación de Plataforma** modifica su **Certificado**.

7.5.4 Lista de Revocación del cliente Operación de Plataforma

La Lista de Revocación del cliente **Operación de Plataforma** se define en la cláusula 5.3 utilizando la asignación de identificador del Cuadro 5.2-2. Los campos entity_id de la Lista de Revocación hacen referencia al campo cosignature_id de la estructura de datos de la firma del cliente **Operación de Plataforma**.

El número mínimo de versión de la Lista de Revocación se define en la inicialización del **Cliente ECI** y se valida utilizando el Sistema de Seguridad Avanzada.

7.5.5 Cofirma del cliente Operación de Plataforma

La cofirma del cliente **Operación de Plataforma** proporciona la firma de la **Operación de Plataforma** a fin de verificar que una imagen de Cliente puede proporcionar servicios de acceso en una plataforma. Además, proporciona la ID del suministrador y del cliente de la imagen para establecer fácilmente la eventual concordancia con la imagen de cliente asociada. Las firmas de clientes **Operación de Plataforma** tienen su propio mecanismo de enumeración de identificadores; ello permite revocar de forma independiente **Imágenes de Cliente ECI** anteriormente permitidas utilizando la Lista de Revocación de cliente **Operación de Plataforma**. En el Cuadro 7.5.5-1 se ofrece información adicional.

Cuadro 7.5.5-1 – Definición de la cofirma del cliente Operación de Plataforma

Sintaxis	N.° de bits	Mnemónico
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
reserved	28	
ECI_Vendor_Id **vendor_id**	32	
if (/* cofirma de series de imágenes */) {		
ECI_Client_Series_Id **client_series_id**	32	
format_version	8	uimsbf
if (format_version == 0x01){		
ECI_Signature_v1 **series_cosignature**		
}		
}		
if (/* cofirma de imagen */){		
ECI_Client_id **client_id**	32	
ECI_Data_Signature **image_cosignature**		
}		
}		

Semántica:

type: byte	Valor de conformidad con el Cuadro 5.2-2.
entity_id: entero	**Identificador** único asignado a la firma en el contexto del **Certificado de Operación de Plataforma**. Se asigna a una sola **imagen de cliente permitida** conjuntamente con el campo cosignature_version.
version: entero	Aumenta su valor (por ejemplo, incrementando los bits más significativos) si la **Operación de Plataforma** modifica su clave pública. Los bits menos significativos de este campo pueden utilizarse para representar (parte de) la versión de las **Series de Imágenes** de cliente o la imagen de cliente por conveniencia de la revocación de gestión de la **Operación de Plataforma** mediante la versión de cliente utilizando el campo versión de la Lista de Revocación del cliente **Operación de Plataforma**.
cosignature_id: ECI_PO_Cosignature_Id	Identificación del identificador de la cofirma en una imagen de cliente. Este campo se incluye en el cálculo de la cofirma.
client_tag: entero	Identificador de forma abreviada con fines de instalación utilizado para designar un client_type en el contexto de una **Operación de Plataforma**. Sólo aquellos clientes que pueden sustituirse mutuamente desde una perspectiva de Usuario tendrán el mismo valor de client_tag. Normalmente, las versiones menores de un cliente son equivalentes.
vendor_id: ECI_Vendor_Id	Id del **Certificado** de suministrador de una **Imagen de Cliente ECI**. Este campo puede utilizarse para localizar **Series de Imágenes** de cliente o una imagen de cliente en la que la cofirma esté incluida en la estructura de datos.
client_series_id: ECI_Client_series_id	Id del **Certificado** de Series de cliente para la verificación de una imagen. El campo tipo del campo client_series_id corresponderá al tipo hijo de los **Certificados de Operación de Plataforma** para client_image_series, véase el Cuadro 5.2-2, y, por lo tanto, define la selección correcta de las interpretaciones alternativas de la estructura de datos.
format_version	Versión del formato de la definición del **Certificado** que se aplica a la cofirma (véase el Cuadro 5.2-1). Concordará con la definición de la versión del **Certificado** de cliente. El único valor válido definido para este campo es 0x01.
series_cosignature: ECI_Signature_v1	Cofirma mediante la clave secreta de la **Operación de Plataforma** del certificado de client_image_series. Los datos de entrada para calcular la firma serán los mismos que los del certificado client_image_series, sustituyendo en la estructura de datos el client_image_series_id por el cosignature_id y el campo extensión por una extensión de 4 bytes que transporta el campo client_image_series_id original del **Certificado**.
client_id: ECI_Client_Id	Id de la imagen de cliente. El campo tipo del campo client_id corresponderá al tipo hijo de los **Certificados de Operación de Plataforma** para client_image, véase el Cuadro 5.2-2 y, por lo tanto, define la selección correcta de interpretaciones alternativas de la estructura de datos.
image_cosignature: ECI_Data_Signature	Cofirma mediante la clave secreta de la **Operación de Plataforma** de la imagen de cliente. Los datos de entrada para el cálculo de la firma son el campo cosignature_id seguido de los datos del fichero imagen de cliente que constituyen la entrada para el cálculo de la firma de la imagen de cliente tal como se define en 7.6.1.

7.6 Formatos de ficheros

7.6.1 Formato del fichero Imagen de Cliente ECI

Las credenciales del **Cliente ECI** contienen los datos necesarios para verificar la autenticidad de la **TA ECI** de un **Cliente ECI**. Utilizará el formato definido en el Cuadro 7.6.1-1.

Cuadro 7.6.1-1 – Definición de las credenciales de cliente

Sintaxis	N.° de bits	Mnemónico
ECI_Client_Credentials {		
ECI_Certificate_Chain **client_chain**		
if (client_chain.chain_length == 0x1) { /* no hay series de clientes; imagen ordinaria */		
ECI_RL **client_rl**		
}		
ECI_Data_Signature **client_signature**		
}		

Semántica:

header: ECI_Client_Chain_Header	Encabezamiento del fichero cadena de **Cliente ECI**.
client_chain: ECI_Client_Chain	**Cadena de Certificados** para validar una **Imagen de Cliente ECI**, que comienza con la Lista de Revocación Raíz de Suministradores de Seguridad, y que finaliza con los Certificados de Suministradores de Seguridad de **Clientes ECI** no basados en Series de Imágenes, o bien, que finaliza con **Certificados** de series de **Clientes ECI** para **Clientes ECI** basados en Series de Imágenes.
client_rl: ECI_RL	Lista de Revocación de identificadores de **Imágenes de Cliente ECI**.
client_signature: ECI_Data_Signature	Firma para validar la **Imagen de Cliente ECI**, cuya clave pública figura en la **cadena de Cliente ECI**.

El fichero **Imagen de Cliente ECI** se define en el Cuadro 7.6.1-2.

Cuadro 7.6.1-2 – Definición del fichero Imagen de Cliente ECI

Sintaxis	N.° de bits	Mnemónico
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) { /* imagen ordinaria */		
ECI_Client_Id **client_id**	32	uimsbf
}		
if (credentials.client_chain..chain_length == 0x2) { /* imagen de series de imágenes */		
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
Reserved	10	
for (i=0; i<n; i++) {		
client_image_byte	8	uimsbf
}		
}		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para la verificación del formato de los datos que siguen. Toma el valor de los tres caracteres ASCII de 8 bits 'ECI' con. El **Anfitrión ECI** comprobará el valor de este campo para verificar si un fichero **ECI** tiene el formato previsto para la integridad de datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor de la versión actualmente definida es 0x01. El **Anfitrión ECI** ignorará cualquier imagen con un número de versión no reconocible.
credentials: ECI_Client_Credentials	**Credenciales del Cliente ECI** para verificar la autenticidad de la **Imagen de Cliente ECI**.
series_image: Booleano	Imagen de Series no es un campo sino una función que se calcula a partir de las credenciales que indican la presencia de un **Certificado** de series de tipo de **Cliente ECI**.
series_id: ECI_Client_Series_Id	ID de series de **Clientes ECI** de las Series de Imágenes de la imagen que sigue. El **Anfitrión ECI** comprobará el valor antes de cargar la **Imagen de Cliente ECI**.
series_image_id: ECI_Client-series_Image_Id	ID de imagen en Series de Imágenes de la imagen que sigue. El **Anfitrión ECI** comprobará el valor antes de cargar la **Imagen de Cliente ECI**.
client_id: ECI_Client_Id	ID de **Cliente ECI** de la **Imagen de Cliente ECI.** El **Anfitrión ECI** comprobará el valor antes de cargar la **Imagen de Cliente ECI**.
vendor_id: ECI_Vendor_Id	ID de suministrador del Suministrador de Seguridad de la **Imagen de Cliente ECI** tal como se define en la estructura de ECI_Vendor_Id de la cláusula 7.4.2. El **Anfitrión ECI** comprobará este campo antes de cargar una (nueva) **Imagen de Cliente ECI**.
image_encrypted_flag: entero	Bandera que señaliza que la imagen está encriptada. Si el valor de este campo es 0b0 la imagen no está encriptada. Si el valor del campo es 0b1 la imagen está encriptada.
online_flag: entero	Bandera que señaliza si el protocolo utilizado para obtener una clave para desencriptar la imagen requiere interacción en línea con el servidor de aprovisionamiento utilizando una palabra de ocasión ("nonce"). Véase 7.8.3
client_image_byte: byte	Secuencia de bytes que contiene la imagen de cliente.

En el Cuadro 7.6.1-2 la frase "el **Anfitrión ECI** comprobará" significa que el **Anfitrión ECI** verificará que el valor utilizado en la práctica se corresponde con alguno de los valores previsibles.

La firma de la **Imagen de Cliente ECI** se calculará en base a todos los datos del fichero que sigue al campo credenciales.

7.6.2 Datos de la Cadena de Operación de Plataforma

El fichero **Imagen de Cliente ECI** se define en el Cuadro 7.6.2-1.

Cuadro 7.6.2-1 – Definición del fichero Cadena de Operación de Plataforma

Sintaxis	N.° de bits	Mnemónico
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		
}		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para la verificación del formato de los datos que siguen. Toma el valor de los tres caracteres ASCII de 8 bits 'EPC'. El **Anfitrión ECI** comprobará el valor de este campo para verificar si un fichero **ECI** tiene el formato previsto para la integridad de datos adicionales.
Image_header_version: byte	Versión del formato del encabezamiento de la imagen. La versión actualmente definida es 0x01. El **Anfitrión ECI** ignorará cualquier imagen con un número de versión no reconocible.
operation_chain: ECI_Client_Chain	**Cadena de Certificados** para validar una **Imagen de Cliente ECI**, que comienza con la Lista de Revocación de la Raíz del Operador y finaliza con el **Certificado de Operación de Plataforma**.
po_client_rl: ECI_RL	Lista de Revocación del cliente **Operación de Plataforma** utilizada para validar la cofirmas de imágenes de cliente. Como parte del proceso de verificación de la cofirma, el **Anfitrión ECI** comprobará los identificadores de cofirma (cosignature_ids) de los datos del cliente **Operación de Plataforma** (po_client_data).
client_image_count: entero	Número de estructuras de datos de firma para imágenes de cliente en el bucle siguiente.

En el Cuadro 7.6.2-1 la frase "el **Anfitrión ECI** comprobará" significa que el **Anfitrión ECI** verificará la concordancia entre el valor utilizado en la práctica y los valores previsibles.

7.6.3 Ficheros de datos de revocación

Existen dos tipos de ficheros de datos de revocación en nombre del **Cargador de Cliente ECI**. Ambos ficheros utilizan el formato de ECI_Revocation_Data_File definido en el Cuadro 5.5-2.

El fichero de datos de revocación del **Cliente ECI** utiliza un father_type igual a 0x0 (**Certificado Raíz**) y un sub_type igual al tipo de la Lista de Revocación de Suministrador. Los revocation_data tienen la restricción de que la Lista de Revocación de hojas de los árboles son listas de revocación de **Cliente ECI**.

El fichero de datos de revocación de la **Operación de Plataforma** utiliza un father_type igual a 0x0 (**Certificado Raíz**) y un sub_type igual al tipo de la Lista de Revocación de **Operador**. Los revocation_data cumplen la restricción de que la Lista de Revocación de hojas de los árboles son listas de revocación de la **Operación de Plataforma**.

7.7 Protocolos de transporte de recursos del Cliente ECI

7.7.1 Generalidades y establecimiento del perfil

En esta cláusula se define la aplicación de protocolos en los CPE y las **Operaciones de Plataforma**.

El protocolo de difusión no proporciona la opción de **Series de Imágenes**. Las imágenes basadas en series solo están previstas en dispositivos conectados mediante el protocolo IP.

Para reducir la carga de tráfico en línea un **CPE** que permita el acceso en modo difusión y en línea a recursos de **Clientes ECI** utilizará prioritariamente el acceso en modo difusión (salvo que en esta Recomendación se indique lo contrario), aunque puede utilizar el acceso en línea en casos de urgencia (**Usuario** en espera), y utilizará el acceso en línea si la red de difusión no dispone de un número mínimo de frecuencias de acceso.

7.7.2 Protocolo de transporte de difusión

7.7.2.1 Introducción

La **ECI** requiere datos de apoyo para varias funciones en nombre del **Cliente ECI** y/o del **Anfitrión ECI** a fin de inicializar y soportar al **Cliente ECI**. Todos los tipos de datos utilizan el mismo protocolo de transporte para todo tipo de datos, tal como se define en esta cláusula. Está estrechamente relacionado con el protocolo utilizado para descargar los ficheros **Imagen de Anfitrión ECI**.

Para su difusión, los datos se organizan en colectores mediante una función hash aplicada al índice de acceso utilizado por el **CPE** a fin de determinar si este necesita los datos. Gracias al uso de colectores, se reduce significativamente el volumen de datos que el CPE debe descargar y se mejora la selectividad en cuanto a los cambios detectados por la supervisión de datos que realmente son pertinentes para el **CPE**.

Se establecen los siguientes grupos de carrusel diferenciados (por tipo de contenido):

- Imágenes de Clientes ECI (por Suministrador de Seguridad).
- Datos de revocación de **Cliente ECI**, organizados en colectores en función de los índices <client_id,client-version_major> y vendor_id.
- **Cadena de Certificados** de operación de plataforma.
- Datos de revocación de **Operación de Plataforma**, organizados en colectores en función de los índices provider_id y operator_id.
- Datos de revocación de **Imagen de Anfitrión ECI**, organizados en colectores.
- Datos de inicialización de la configuración de Seguridad avanzada (AS_setup) **ECI** de **Cliente ECI,** organizados en colectores.
- Grupos de carruseles definidos para estructuras de datos de importación y exportación (véase 9.8).
- Grupos de carruseles definidos para datos propiedad del **Operador**.

Todos los parámetros del carrusel DSMCC cumplirán lo establecido en [ETSI EN 301 192].

Un **Operador** puede utilizar varios carruseles en múltiplex separados para transmitir todos los datos necesarios. No obstante, para un **Cliente ECI** específico el **Anfitrión ECI** sólo tendrá que supervisar las actualizaciones de una única DII de ubicación de un carrusel de datos.

7.7.2.2 Traspaso de credenciales y datos de revocación al Operador

Los formatos de datos y los protocolos para la transferencia de credenciales y de listas de revocación a un **Operador** no forman parte de la especificación **ECI**.

7.7.2.3 Traspaso del Suministrador de Seguridad al Operador

Los formatos de datos y los protocolos para la transferencia de contenido del **Suministrador de Seguridad** al **Operador** no forman parte de la presente Recomendación.

7.7.2.4 Señalización PSI

Los carruseles utilizarán el stream_identifier_descriptor [ETSI EN 300 468] en la PMT a fin de etiquetar el flujo utilizado para transmitir el carrusel al objeto de permitir que se establezcan referencias mediante el descriptor data_broadcast en la información del servicio (SI).

Los carruseles utilizarán un data_broadcast_id_descriptor con data_broadcast_id tal como se define en el Cuadro 7.7.2.4-1.

Cuadro 7.7.2.4-1 – Valor del ID de difusión de datos para carruseles específicos de la ECI

Valor de Data_broadcast_id	Significado
Atribuido por la oficina de proyecto DVB, véase el valor del identificador de difusión (broadcast-id) definido en [ETSI TS 101 162].	El cliente específico del **Operador ECI** soporta el carrusel de datos.

Los bytes selectores del data_broadcast_id_descriptor tendrán la estructura que se define en el Cuadro 7.7.2.4-2.

Cuadro 7.7.2.4-2 – Estructura del ID de carrusel para carruseles de datos DSMCC DVB ECI

Sintaxis	N.° de bits	Mnemónico
ECI_carousel_id_structure {		
version	8	uimsbf
if (version == 0x01){		
operator_id	20	uimsbf
platform_**operation_id**	20	uimsbf
}		
}		

Semántica:

version: entero	Versión de la estructura; actualmente sólo está definido el valor 0x01. Los restantes valores están reservados. Los **CPE** que encuentren una versión distinta a 0x01 ignorarán este descriptor.
operator_id: ECI_Operator_Id	ID de la **ECI** del **Operador** (definido para cualquier **Certificado de Operador**) de la **Operación de Plataforma** del carrusel.
platform_operation_id: ECI_Platform_Operation-Id	Conforme al **Certificado de Operación de Plataforma**: ID de la **Operación de Plataforma**.

7.7.2.5 Señalización de información del servicio (SI)

7.7.2.5.1 Señalización de ubicación del carrusel de datos mediante el descriptor de vinculación de la ubicación de datos

El descriptor de vinculación de la ubicación de datos del **Cliente ECI** es un descriptor de vinculación DVB privado de la **ECI** [ETSI TS 101 162]. Este descriptor de vinculación ayuda a un **CPE** a determinar la ubicación del múltiplex que transporta un carrusel de datos de **Cliente ECI** para una **Operación de Plataforma** específica. La NIT o la BAT transportan este descriptor de vinculación. El descriptor de vinculación de la ubicación de datos del **Cliente ECI** siempre estará precedido en la sección de la tabla por un descriptor que especifique datos privados DVB [ETSI TS 101 162] cuyo valor del campo private_data_specifier sea "ECI", tal como se define en [ETSI TS 101 162]. Este descriptor puede aparecer varias veces en la NIT o la BAT. Este descriptor de vinculación será transportado en redes y en paquetes de programas con más de 4 multiplexores.

Con relación a la definición del descriptor de vinculación definido en [ETSI EN 300 468] y [ETSI TS 101 211] los campos del descriptor de vinculación de la ubicación de datos del **Cliente ECI** tienen la aplicación específica siguiente:

- **service_id**: puede tomar el valor 0x0000 para indicar que no se señaliza un service_id en particular.
- **linkage_type**: valor 0x80 que señaliza un descriptor de vinculación de la ubicación de datos de **Cliente ECI**.

El campo de bytes de datos privados del descriptor de vinculación de la ubicación de datos del **Cliente ECI** tendrá la estructura definida en el Cuadro 7.7.2.5.1-1.

Cuadro 7.7.2.5.1-1 – Estructura de datos privados del descriptor de vinculación de la ubicación del carrusel de datos del Cliente ECI

Sintaxis	N.° de bits	Mnemónico
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
}		
}		
}		

Semántica:

version: entero	Versión de la estructura; actualmente sólo está definido el valor 0x01. Los restantes valores están reservados. Los **CPE** que encuentren una versión distinta a 0x01 ignorarán este descriptor.
operator_id: ECI_Operator_Id	ID de la **ECI** del **Operador** (definido para cualquier **Certificado de Operador**) de la **Operación de Plataforma** del carrusel. El valor 0x00000 señaliza a cualquier **Operador**.
platform_operation_id: ECI_Platform_Operation-Id	Conforme al **Certificado de Operación de Plataforma**: ID de la **Operación de Plataforma**. El valor 0x0000 señaliza a cualquier **Operación de Plataforma**.

Los operadores de red y de paquetes de programas pueden utilizar especificadores comodín (de valor 0x00000) para el operator_id o el platform_operation_id a fin de establecer un vínculo con un múltiplex que transporta uno o más carruseles de datos de **Cliente ECI**. Por motivos de eficiencia se recomienda que dicha señalización se limite a ayudar a los **CPE** a inspeccionar el número mínimo de multiplexadores necesarios para localizar un carrusel de **Operación de Plataforma** específico.

Se recomienda utilizar un único descriptor de vinculación de ubicación de carrusel de datos de **Cliente ECI** con un múltiplex en una NIT o BAT, y que todos los carruseles aplicables ubicados en ese múltiplex se enumeren en una única estructura ECI_Client_data_location.

7.7.2.5.2 Descriptor de descarga de emergencia de Cliente ECI

La necesidad de sustituir urgentemente una **Imagen de Cliente ECI** puede indicarse colocando uno o varios descriptores ECI_client_emergency_download en la NIT, la BAT o en una de las entradas de la SDT para un servicio accesible gracias a dicho **Cliente ECI**. El **Anfitrión ECI** podrá obtener este descriptor de cualquiera de las tablas en las que aparezca en cualquiera de los múltiplex sintonizados, realizar el procesamiento asociado y utilizar un sintonizador de reserva para acceder a los múltiplex pertinentes al objeto de adquirir el descriptor a intervalos máximos de 30 minutos.

El ECI_client_emergency_download_descriptor permite dirigirse a plataformas de operación y tipos de anfitrión específicos a fin de minimizar las perturbaciones causadas por actualizaciones de emergencia.

Cuando el **Anfitrión ECI** encuentra un nuevo descriptor ECI_client_emergency_download (verificado mediante la tabla de origen y el campo emergency_id), comparará la configuración de su Anfitrión y su Cliente con la información objetivo del descriptor. Si se detecta concordancia de objetivos y la versión de la imagen de cliente instalada requiere actualización, el Anfitrión realizará la actualización conforme al emergency_indicator. En caso de un conflicto de recursos, pueden interrumpirse las actividades en curso del **Usuario** en el **CPE**.

El descriptor de operación **ECI** es un descriptor privado DVB que siempre estará precedido en la tabla en la que aparezca por el private_data_specifier_descriptor DVB utilizando el private_data_specifier_field de la **ECI** (véase [ETSI EN 300 468]). En el Cuadro 7.7.2.5.2-1 se define la sintaxis del descriptor.

Cuadro 7.7.2.5.2-1 – ECI_Client_Emergency_Download_Descriptor

Sintaxis	N.° de bits	Mnemónico
ECI_client_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* bucle principal */		
main_loop_nr	8	uimsbf
for (i=0; i<main_loop_nr; i++){		
/* platforma objetivo */		
platform_operation_tag	8	uimsbf
/* bucle objetivo de cliente */		
host_nr	8	uimsbf
/* bucle objetivo de id de anfitrión */		
for (j=0; j<host_nr; j++){		
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
host_version	8	uimsbf
}		
/* bucle de imagen de cliente */		
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
}		
}		
/* datos privados hasta el final del descriptor */		
for (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Semántica:

descriptor_tag	Valor de la etiqueta privada **ECI** para el descriptor_tag: véase [b-UIT-T J Supl. 7].
descriptor_length	Véase [ETSI EN 300 468].
main_loop_nr	Número de entradas en el bucle principal. Cada una de las entradas del bucle principal será analizada por separado por el **Anfitrión ECI**, es decir, tienen una semántica OR. Los diversos elementos de una misma entrada del bucle tendrán una semántica AND.
platform_operation_tag	Valor de la etiqueta para la plataforma **ECI** tal como se enumera en el ECI_platform_operation_descriptor en la NIT/BAT. El **Anfitrión ECI** considerará que debe haber una actualización de emergencia si la platform_operation concuerda con el platform_operation de uno de los **Clientes ECI** instalados.
host_nr	Número de entradas en el bucle objetivo del anfitrión; el valor 0 significa que todos los **Anfitriones ECI** son objetivo. Las entradas del bucle tendrán una semántica OR; es decir, si alguna especificación objetivo de anfitrión es concordante, el estado de la condición objetivo del bucle principal será de concordancia.
manufacturer_id	Manufacturer_id del anfitrión objetivo de una actualización de emergencia. El anfitrión considerará una actualización de emergencia si el valor de este campo concuerda con el manufacturer_id del **Anfitrión ECI**.
cpe_type_id	Valor según define el ECI_CPE_Type_ID en el Cuadro 6.2.2.1-2. El **Anfitrión ECI** considerará una actualización de emergencia si el cpe_type_id del anfitrión concuerda con el valor de este campo. Si cpe_type_id.cpe_type es 0x000 significa que existe concordancia de cualquier cpe_types de **Anfitrión ECI** (y se ignorarán el cpe_model y la versión del anfitrión). Si cpe_type_id.cpe_model es 0x00 significa que existe concordancia de cualquier cpe_model de **Anfitrión ECI** (y se ignorará la versión del anfitrión).
host_version	El **Anfitrión ECI** considerará una actualización de emergencia si y solo si su versión de anfitrión es menor o igual al valor de este campo. Véase la Nota.
client_nr	Número de entradas del bucle de imágenes de cliente. Las entradas del bucle tendrán una semántica OR y todos los clientes para los que exista concordancia serán candidatos para una actualización de emergencia.
emergency_indicator	El **Anfitrión ECI** utilizará el valor de este campo para seleccionar el comportamiento adecuado a fin de iniciar la descarga y ulterior actualización del cliente como se define en el Cuadro 7.7.2.5.2-2.
client_tag	Valor de la etiqueta que identifica el **Cliente ECI** tal como se enumera en el ECI_platform_operation_descriptor en la NIT/BAT que concuerde con el campo platform_operation_tag del mismo bucle principal. El **Anfitrión ECI** considerará una actualización de emergencia si los vendor_id y client_id concuerdan con alguno de los clientes instalados en el **Anfitrión ECI**.
min_client_version_major	Este campo representa el número de versión superior mínimo aceptable para la imagen de cliente. El **Anfitrión ECI** considera una actualización de emergencia si un Cliente se instala y existe concordancia con client_tag cuyo número de versión superior sea inferior al valor de este campo.
min_client_version_minor	Este campo representa el número de versión menor mínimo aceptable para la imagen de cliente. El **Anfitrión ECI** considera una actualización de emergencia si un **Cliente ECI** se instala y existe concordancia con client_tag cuyo número de versión menor sea inferior al valor de este campo y la versión superior sea igual a min_client_version_major.
client_id	Identificador de cliente de un **Cliente ECI** que proporciona servicios de desencriptación para servicios con platform_operation_tag, tal como se define en el Cuadro 7.4.4-1.
private_data_byte	Datos privados: el contenido puede ser definido por el **Operador** que gestiona la difusión de este descriptor.
NOTA – Un valor del campo igual a 0xFF implica la concordancia de todas las versiones de anfitrión.	

En el Cuadro 7.7.2.5.2-1 se define un conjunto de condiciones del bucle principal (que tiene una semántica AND) que deberán cumplirse para que el **Anfitrión ECI** considere realizar una actualización de emergencia. Si se cumplen todas esas condiciones, el **Anfitrión ECI** realizará la descarga e instalación de emergencia de una o más imágenes de cliente de conformidad con el campo emergency_indicator de ese cliente.

Cuadro 7.7.2.5.2-2 – Valores del campo emergency_indicator del ECI_Client_emergency_download_descriptor

Nombre	Valor	Descripción
Emergencia del sistema	0x01	El **Anfitrión ECI** descargará la nueva imagen de cliente y la instalará tan rápidamente como sea posible interrumpiendo, si es preciso, actividades en curso iniciadas por usuarios (véase la Nota 1).
Emergencia del cliente	0x02	El **Anfitrión ECI** descargará la nueva imagen de cliente y la instalará antes de la apertura de alguna sesión de distintivo de medios para ese Cliente. En primer lugar se terminará cualquier sesión en curso del distintivo de medios para ese Cliente (véase la Nota 2).
Urgencia del cliente	0x03	El **Anfitrión ECI** descargará la nueva imagen de cliente y la instalará en la primera ocasión en que ello no interrumpa actividad alguna iniciada por un **usuario.** El **Anfitrión ECI** descargará la nueva imagen del anfitrión a más tardar durante el siguiente evento de encendido (véase la Nota 3).
RFU	otros	Reservados para uso futuro.
NOTA 1 – Los **Operadores** pueden utilizarlo, por ejemplo, si el **Cliente ECI** actual puede dañar al **Anfitrión ECI** y/o otros **Clientes ECI** y debe ser sustituido inmediatamente. NOTA 2 – Los **Operadores** pueden utilizarlo, por ejemplo, si los servicios de desencriptación del **Cliente ECI** actual tienen una calidad de funcionamiento muy deficiente. NOTA 3 – Los **Operadores** pueden utilizarlo, por ejemplo, si el **Cliente ECI** actual presenta serias deficiencias en relación con la desencriptación de servicios, pero funciona razonablemente bien en los casos de uso habituales.		

7.7.2.6 Descriptor de compatibilidad del carrusel

En los mensajes DII de DSI se utilizará el compatibilityDescriptor que emplean los carruseles de datos DMSCC DVB [ETSI EN 301 192].

El compatibilityDescriptor proporciona información sobre el tipo de datos transportados en un grupo de carruseles. El specifierData() contiene el OUI **ECI**. En el Cuadro 7.7.2.6-1 se definen los campos aplicables del compatibilityDescriptor en los carruseles de datos del **Cliente ECI**.

Cuadro 7.7.2.6-1 – Tipos de contenido del carrusel de datos ECI

Campo tipo de descriptor	Finalidad del grupo	Campo Modelo	Campo versión	Índice colector para calcular el ID de módulo
0xA0	**Imágenes de Cliente ECI** y ficheros de credenciales para un Suministrador	Vendor_id del **Suministrador de Seguridad** de las imágenes		Asignación libre
0xA2	Ficheros de datos de revocación del **Cliente ECI** (como colectores)	platform_operation_id		= Vendor_id + <Client_type, client_version_major> (véase la Nota)
0xA3	Fichero de la cadena de **Operación de Plataforma**	platform_operation_id, platform_operation_version		Asignación libre
0xA4	Ficheros de datos de revocación de **Operación de Plataforma** (como colectores)	platform_operation_id		= Operator_id + provider_id
0xA5	Ficheros de datos de revocación del **Anfitrión ECI** (como colectores)	platform_operation_id		= Manufacturer_id + cpe_type_id
0xA6	Ficheros AS_setup (como colectores)	platform_operation_id		target_id para el **CPE**
0xA7-0xAA	Contenedor de aplicación UI (véase 9.4.3.4.2)	Definido por el **Operador**		Asignación libre
0xB0	Fichero de árbol de exportación	platform_operation_id (el **ECI Cliente** exportador)		Asignación libre
0xB1	Fichero de cadenas de importación	platform_operation_id (del **Cliente ECI** importador)		Asignación libre
0xB2	Fichero de cadenas de autenticación de importación	platform_operation_id (de **Cliente ECI** importador)		Asignación libre
0xB8-0xBF	Formato propiedad del operador	Definido por el **Operador**		Definido por el **Operador**
Otros valores	reservados			
NOTA – Concatenación de los dos campos, con el más significativo como primer argumento y la generación de un número de 20 bits.				

El cálculo del índice del colector utiliza una aritmética con valores enteros de módulo 32 bits, definida en 7.7.2.7.

7.7.2.7 DSI del carrusel

Si el carrusel tiene dos capas, el DSI incluirá un índice completo de los grupos del carrusel (es decir, una entrada de bucle por cada DII).

El compatibilityDescriptor se define en el Cuadro 7.7.2.6-1. Los campos de indicación de información de descarga (DII) no incluidos en el bucle cumplirán las limitaciones siguientes:

- Tamaño del bloque: al menos 512 bytes, para grupos con módulos mayores se recomienda al menos 2 kbytes.
- tCDownloadScenario: será al menos 4 veces el mensaje de repetición DDB más lento del grupo. TCDownload también cumplirá las restricciones máximas incluidas en el Cuadro B.4-1.
- numberOfModules: refleja el número de módulos para carruseles ordinarios y el número de colectores (cada uno corresponde a un módulo) para datos organizados en colectores. Para los datos de una **Cadena de Certificados de Operación de Plataforma** el valor será 1.

Los valores de tCDownloadScenario que se muestran más delante reflejan el plazo de temporización para que un **CPE** adquiera un elemento de datos completo. Será como mínimo cuatro veces el tiempo de repetición DDB más lento de cualquiera de los módulos del grupo. En la cláusula B.4 se definen los valores correspondientes a los distintos elementos.

Los siguientes campos de bucle de módulo cumplirán las restricciones indicadas a continuación:

- moduleId: los bits 15 a 8 coincidirán con los bits menos significativos (LSB) del groupId en la correspondiente estructura groupInfo del DSI. Los bits 7 a 0 se asignan con arreglo al Cuadro 7.7.2.7-1.
- moduleVersion: la aplicación depende del tipo de carrusel, y será conforme con el Cuadro 7.7.2.7-1.
- moduleInfoLength: 0 para todos los carruseles **ECI**.

Cuadro 7.7.2.7-1 – Parámetros de los grupos del carrusel ECI

Tipo de grupo	Bit 7..0 del Id del módulo	Versión del módulo	Información del módulo
Imágenes de cliente	tipo de cliente (client_type)	Versión de cliente (client_version)	Ninguna
Datos de revocación de cliente	número de colector (bucket_number)	Se incrementa con cada actualización	Ninguna
Cadena de clientes de **Operación de Plataforma**	Asignado por el **Operador**	Se incrementa con cada actualización	Ninguna
Datos de revocación de **Operación de Plataforma**	número de colector (bucket_number)	Se incrementa con cada actualización	Ninguna
Datos de revocación de **Anfitrión ECI**	número de colector (bucket_number)	Se incrementa con cada actualización	Ninguna
Datos de configuración de AS (AS_setup) de la ECI	número de colector (bucket_number)	Se incrementa con cada actualización	Ninguna

Para los números organizados según los colectores, el número de colector (igual al bit [7..0] del module_id) se calculará a partir del índice con una operación de módulo simple:

```
bucket_number = bucket Index % numberOfModules
```

7.7.2.8 DDB del carrusel

No existen requisitos específicos.

7.7.2.9 Comportamiento dinámico del carrusel

La actualización de la numeración de la versión del carrusel y de DSI y DII serán conforme a [ETSI TR 101 202]. Ello implica que cualquier actualización de un módulo quedará reflejada en el número de versión del módulo, su DII y una progresión en cascada ascendente hacia el DSI (si existe).

La implementación del **CPE** puede supervisar los cambios que se produzcan en sus módulos objetivos para el seguimiento de cualquier actualización dinámica durante el funcionamiento normal.

7.7.3 Protocolos de transporte en Internet

7.7.3.1 Introducción

El **Anfitrión ECI** puede obtener los diversos elementos de datos requeridos de un servidor designado por el **Operador**.

La interfaz utilizará peticiones HTTPS directas tal como se especifica en la cláusula 9.4.4.6 de acuerdo con los principios de diseño RESTfull [b-Richardson] codificando la petición como una combinación de extensión URL y parámetros de consulta, con la contestación codificada como un fichero binario.

El servidor HTTP responderá con uno de los siguientes códigos de estado:

- 200: OK (se devuelve el fichero solicitado).
- 302 FOUND: redicrección de la petición a otro servidor; la petición http se repite en el URL devuelto.
- 404: Elemento no presente en el servidor.
- 500 .. 599: Error de servidor.

La especificación de los URL utilizados en las peticiones emplea la notación "Bachus Naur". Los nombres de los símbolos correspondientes a campos en las estructuras de datos **ECI** representarán su valor en hexadecimal (cadena de caracteres '0' .. '9' , 'A' .. 'F'), con un número de dígitos doble utilizados como bytes para representar el número en las estructuras de datos binarios internas **ECI**. El servidor ignorará cualquier parámetro de consulta adicional que no reconozca.

7.7.3.2 Visión general de la API web de la ECI

El **Operador** soportará un servidor en línea que responda a la petición HTTP1.1 [IETF RFC 7231] GET que tenga la sintaxis y semántica URL siguiente:

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

donde **major** y **minor** reflejan el número superior y menor de la versión del protocolo en una representación decimal sin ceros iniciales. La versión actual es 1.0. En el Cuadro 7.7.3.2-1 se presenta la definición de cola.

Cuadro 7.7.3.2-1 – Definición de cola

```
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*.
```

El valor de tail_extension indica diversas opciones de extensión de la API web de la **ECI** definidas en la presente Recomendación.

7.7.3.3 Peticiones de la API web relacionadas con el Anfitrión ECI

Se definen las siguientes peticiones de la API web relacionadas con el **Anfitrión ECI**:

- host_version ::= 'host-version ' '?target-id=' target_id.
  Devolverá la última versión del conjunto **Imagen de Anfitrión ECI** para el **CPE** identificado por **target_id**.
- host_images ::= 'hi-images' ' ?target-id=' target_id.
  Devolverá el último número de imágenes de un **Anfitrión ECI** para el **CPE** identificado por **target_id**.
- host_image_version ::= 'hi-version' ' ?target-id=' target_id '&image-id=' image_id .
  Devolverá la última versión del **id_image** del fichero **Imagen de Anfitrión ECI** para el **CPE** identificado por **target_id**.
- host_image ::= 'host-image' ' ?target-id=' target_id '&image-id=' image_id.
  Devolverá el último **image_number** de la **Imagen de Anfitrión ECI** para el **CPE** identificado por **target_id**. image_number=="FF" devuelve el fichero de credenciales del **Anfitrión ECI** para las **Imágenes de Anfitrión ECI**, incluidos los datos de revocación más recientes.

En el caso de las peticiones relacionadas con el **Anfitrión ECI**, el servidor de una **Operación de Plataforma** puede soportar **Anfitriones ECI** para cualquier tipo de **CPE** que desee. Si soporta un tipo de **CPE** también soportará el conjunto completo más reciente de **Imágenes de Anfitrión ECI** y las correspondientes consultas sobre **host_image_version**, **host_images** y **host_revocation**. El formato del fichero que se devuelve es ECI_Host_Version_File definido en el Cuadro 7.7.3.3-1.

Cuadro 7.7.3.3-1 – Definición del fichero versión de Anfitrión ECI

Sintaxis	N.° de bits	Mnemónico
ECI_Host_Version_File {		
magic = 'RHVE'	32	uimsbf
host_version	8	uimsbf
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RHIM' con caracteres ASCII de 8 bits.
host_version: entero	Número de versión del **Certificado de Anfitrión ECI**.

El formato del fichero devuelto es ECI_Host_Images_File definido en el Cuadro 7.7.3.3-2.

Cuadro 7.7.3.3-2 – Definición del fichero Imágenes de Anfitrión

Sintaxis	N.° de bits	Mnemónico
ECI_Host_Images_File {		
magic = 'RHIM'	32	uimsbf
host_images	8	uimsbf
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RHIM' con caracteres ASCII de 8 bits.
host_images: entero	Número de **Imágenes de Anfitrión ECI** soportado por el tipo de **CPE** identificado en la petición.

El formato del fichero devuelto es ECI_Host_Image_Version_File definido en el Cuadro 7.7.3.3-3.

Cuadro 7.7.3.3-3 – Sintaxis del fichero versión de la Imagen de Anfitrión

Sintaxis	N.° de bits	Mnemónico
ECI_Host_Image_Version_File {		
magic = 'RHIV'	32	uimsbf
host_image_version	16	uimsbf
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RHIV' con caracteres ASCII de 8 bits.
host_image_version: entero	Versión de **Imagen de Anfitrión ECI** de la **Imagen de Anfitrión ECI** identificada en la petición.

7.7.3.4 Peticiones de la API web relacionadas con la Operación de Plataforma

El servidor de la **Operación de Plataforma** admite las peticiones siguientes en nombre del id de **Operación de Plataforma** permitido:

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

Devolverá el estado de revocación del **Certificado** emitido para el **operator_id**, **platform_operation_id** en el formato de fichero definido en el Cuadro 7.7.3.4-1. El servidor para una **Operación de Plataforma** soportará como mínimo sus propios **Certificados de Operación de Plataforma** en funcionamiento a través de la interfaz.

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

Devolverá el estado de revocación de la plataforma de la **Imagen de Cliente ECI** para el cosignature_id conforme a la última Lista de Revocación de cliente operación de plataforma. Véase el Cuadro 7.7.3.4-2.

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

Devolverá la última cadena del **Cliente ECI** para la **Operación de Plataforma** identificada por el **operator_id**, **platform_operation_id**, tal como se define en el Cuadro 7.6.2-1. El servidor para una **Operación de Plataforma** admitirá como mínimo sus propios certificados de **Operación de Plataforma** en funcionamiento a través de la interfaz.

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

Devolverá el último fichero de datos de revocación de la **Operación de Plataforma** que contiene la Lista de Revocación del **Operador** identificado por **operator_id**. El servidor admitirá como mínimo los últimos datos de revocación para el **Operador** de su propia **Operación de Plataforma**. Los **Anfitriones ECI** utilizarán esta API para intentar adquirir los datos de revocación más recientes de todos los **Clientes ECI** almacenados.

Cuadro 7.7.3.4-1 – Sintaxis del fichero verificación de Operación de Plataforma

Sintaxis	N.° de bits	Mnemónico
ECI_PO_Check_File {		
magic = 'RPCH'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RHIV' con caracteres ASCII de 8 bits.
non_revoked_certificate_flag: byte	Toma el valor 0x00 si se ha revocado el **Certificado** del ID de **Operación de Plataforma** identificado por la petición, o bien 0x01 en cualquier otro caso.

Cuadro 7.7.3.4-2 – Sintaxis del fichero de verificación de Cliente Operación de Plataforma

Sintaxis	N.° de bits	Mnemónico
ECI_PO_Client_Check_File {		
magic = 'RPCC'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RHIV' con caracteres ASCII de 8 bits.
non_revoked_certificate_flag: byte	Toma el valor 0x00 si la imagen de cliente asociada con el campo cosignature_id de la petición ha sido revocada previamente con arreglo a la última lista de Revocación de Cliente Operación de Plataforma de la Operación de Plataforma, o bien 0x01 en cualquier otro caso.

7.7.3.5 Peticiones de cliente de la API web

El servidor del **Operador** soportará las peticiones siguientes en nombre de los clientes que son requeridas para su id de **Operación de Plataforma**.

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

- Devolverá un Fichero Versión de Cliente (véase el Cuadro 7.7.3.5-1) que contiene la última versión de la **Imagen de Cliente ECI** para un cliente identificado por **vendor_id** y **client_type**. El servidor admitirá como mínimo los clientes utilizados para operar sus propios servicios de **Operación de Plataforma**.

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

- Devolverá un Fichero Versión de Credencial de Cliente (véase el Cuadro 7.7.3.5-2) que contiene la última versión de las Credenciales del **Cliente ECI** para un cliente identificado por **vendor_id** y **client_type**. El servidor soportará como mínimo los clientes utilizados para operar sus propios servicios de **Operación de Plataforma**.

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

- Devolverá el último fichero **Imagen de Cliente ECI** para un cliente identificado por <vendor_id, client_type, client_version_major>. En el caso de una **Imagen**, se proporciona como parámetro de consulta un identificador image_target_id de tipo ECI_Image_Target_Id. El servidor soportará como mínimo los **Suministradores** de **Clientes ECI** utilizados para operar sus propios servicios de **Operación de Plataforma**. Los **Anfitriones ECI** utilizarán esta API para intentar adquirir los datos de revocación más recientes de todos los **Clientes ECI** almacenados.

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

- Devolverá el fichero de datos de revocación de **Cliente ECI** más reciente de un cliente identificado por **vendor_id**. El servidor admitirá como mínimo los clientes utilizados para operar sus propios servicios de **Operación de Plataforma**.

Cuadro 7.7.3.5-1 – Sintaxis del fichero versión del cliente

Sintaxis	N.° de bits	Mnemónico
ECI_Client_Version_File {		
magic = 'RCVE'	32	uimsbf
client_version	16	uimsbf
emergency_download_descriptor		
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RCVE' con caracteres ASCII de 8 bits.
client_version: entero	Versión de cliente más reciente del tipo de cliente identificado en la petición.
emergency_download_descriptor	**ECI_client_emergency_download_descriptor** en el que el **Anfitrión ECI** asumirá una etiqueta platform_operation_tag que concuerde con la Operación de Plataforma del proveedor de la API web del cliente y que la etiqueta client_tag concuerde con la imagen de cliente, según se solicite en los parámetros de la api web.

Cuadro 7.7.3.5-2 – Sintaxis del fichero versión de credencial de cliente

Sintaxis	N.° de bits	Mnemónico
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
}		

Semántica:

magic: byte[4]	Representación de la cadena 'RCCV' con caracteres ASCII de 8 bits.
root_version: entero	Versión raíz (como se define en el Cuadro 5.3-1) de las credenciales de **Cliente ECI** más recientes.
vendor_rl_version: entero	Número de versión de la Lista de Revocación de Suministrador de Seguridad de las credenciales de **Cliente ECI** más recientes.
eci_vendor_id: ECI_Vendor_Id	ECI_Vendor_Id (como se define en el Cuadro 7.6.1-2) de las credenciales de **Cliente ECI** más recientes.
client_rl_version: entero	Número de versión de la Lista de Revocación de cliente de las credenciales de **Cliente ECI** más recientes.
eci_client_id: ECI_Client_Series-Id	ECI_Client_Series_Id (como se define en el Cuadro 7.6.1-2) de las credenciales de **Cliente ECI** más recientes.

7.7.3.6 Peticiones AS_setup de la API web

Si el **Operador** admite el registro en línea de **Clientes ECI** en modo encriptado, también se admitirá la petición siguiente:

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

La respuesta a la petición es el fichero as_setup para el cliente especificado (<vendor_id,eci_client_id>) y para el **CPE** especificado por ECI_Image_Target_Id target_id. El tipo de eci_client_id puede ser ECI_Client_Id o ECI_Client_Series_Id. El valor de la palabra de ocasión es "Nonce" ("palabra de ocasión") tal como se especifica en el protocolo de desencriptación de **Imagen de Cliente ECI**. Para más información véase la cláusula 7.8.4.2.

7.8 Instalación del Cliente ECI Operación de Plataforma

7.8.1 Alcance y establecimiento del perfil

La **Operación de Plataforma** puede seleccionar las opciones de seguridad para las instalaciones del **Cliente ECI** y señalizarlas utilizando image_encrypted_flag y la bandera en línea en el fichero **Imagen de Cliente ECI** (véase el Cuadro 7.6.1-2):

- "modo de instalación **Cliente ECI** con el fichero **Imagen de Cliente ECI** desencriptado", en la que se descarga (la última versión de) el **Cliente ECI** propuesto por la señalización definida en la cláusula 7.2 y se inicializa **Cliente ECI**.
- "modo de instalación de **Cliente ECI** con el fichero **Imagen de Cliente ECI** encriptado", que además del primer modo permite que la **Operación de Plataforma** encripte la **Imagen de Cliente ECI** y la autentique como se define en [UIT-T J.1014]. La desencriptación del **Cliente ECI** es específica del **Anfitrión ECI** e incluye la verificación de la versión del **Anfitrión ECI,** garantizando además la confidencialidad del **Cliente ECI** tras la desencriptación al no permitir la desencriptación de **Anfitriones ECI** desconocidos o en situación comprometida. Si el **CPE** no está conectado a una red en línea es necesario un ECI_Image_Target_Id. En este caso de uso el ECI_Image_Target_Id debe enviarse manualmente a la cabecera responsable de la seguridad.

En el resto de esta cláusula se define el protocolo para ambas versiones de inicialización del **Cliente ECI**.

Las **Operaciones de Plataforma** que utilizan los **CPE** en línea en modo de instalación con encriptación pueden forzar el uso del **Cliente ECI** más reciente utilizando una palabra de ocasión generada por mecanismos de Seguridad Avanzada (AS) en el protocolo de desencriptación con el servidor de la **Operación de Plataforma** para el **Cliente ECI** (véase 7.7.3.6).

Normas para el establecimiento del perfil:

- El **CPE** utilizará el protocolo de registro en línea si la **Operación de Plataforma** ofrece el registro en línea (la señalización se define en 7.2) y el **CPE** pueda acceder a los servicios en línea.
- Los **CPE** que pueden recibir en modo difusión, podrán ejecutar el protocolo de registro por difusión. El modo difusión requiere el registro del **CPE** durante el registro inicial de la **Operación de Plataforma**.
- Las Operaciones de Plataforma que admitan redes de difusión que soporten a los **CPE** sin conectividad simultánea en línea permitirán el registro en modo difusión. La información relativa a la introducción por el **Usuario** de información de registro para un **CPE** seguirá las reglas de formato aplicables.

7.8.2 Modo de instalación de Cliente ECI con un fichero Imagen de Cliente ECI desencriptado

Cuando comienza la inicialización del **Cliente ECI**, el **Anfitrión ECI** reserva un **intervalo AS** para la **Operación de Plataforma**, reinicializa el **intervalo AS** y carga la clave pública de la **Operación de Plataforma** en el **intervalo AS** tal como define [UIT-T J.1104].

Si es necesario, el **Anfitrión ECI** descarga el **Cliente ECI**, lo almacena en RAM NV para recuperarlo más adelante y lo arranca. El **Cliente ECI** guiará al **Usuario** en la instalación. La instalación puede requerir que el **Usuario** envíe manualmente el target_id del ECI_Image_Target_Id del **CPE** a la cabecera en caso de que el **CPE** no disponga de una conexión en línea para el registro de seguridad del sistema de difusión.

En cualquier recarga subsiguiente, el **Anfitrión ECI** reinicializará el **Cliente ECI**.

7.8.3 Modo de instalación del Cliente ECI con un fichero Imagen de Cliente ECI encriptado

Este modo de funcionamiento realiza una descarga encriptada de la **Imagen de Cliente ECI** utilizando una clave seleccionada por el **Operador**. Esta clave seleccionada por el **Operador** se encripta y transporta en una estructura **as_setup**.

Al comienzo de la inicialización del **Cliente ECI**, el **Anfitrión ECI** reserva un **intervalo AS** para la **Operación de Plataforma**, reinicia el intervalo-AS y carga la clave pública de la **Operación de Plataforma** en el **intervalo AS**:

- El **Anfitrión ECI** diferenciará dos modos para la recuperación de **as_setup**: **Modo de registro** se accede a este modo si el **Cliente ECI** se inicia por primera vez o ha cambiado la clave pública de la operación de plataforma (POPK) o la versión del **Cliente ECI**, o bien el cliente funciona en modo de renovación de registro en línea con una palabra de ocasión única para cada renovación de registro. La estructura **as_setup** para el **CPE** se obtendrá de la red de la **Operación de Plataforma**.
- **Modo registrado**: la estructura **as_setup** previa se recupera de la memoria NV. Si hay pendiente algún cambio de versión de **Cliente ECI** o de **Anfitrión ECI**, el **Cliente ECI** debe avisar al **Usuario** para que inicie o desbloquee dicha descarga (en el caso de descarga por defecto, esto debe suceder de forma automática en un plazo de tiempo razonable). La descarga de un nuevo **Cliente ECI** también requerirá una nueva estructura **as_setup**.

En el modo de registro el **Anfitrión ECI** ejecutará las acciones siguientes para la recuperación de una nueva estructura **as_setup**:

1) El **Anfitrión ECI** inicializa el intervalo-AS y obtiene:
   - El valor del target_id del ECI_Image_Target_Id del **CPE**.
   - Una palabra de ocasión ("nonce") (128 bits) obtenida del **intervalo AS** mediante la función **getAsSlotRk** (véase [UIT-T J.1014]) en el caso de registro en línea.
2) El **Anfitrión ECI** enviará la información anterior para obtener un mensaje **as_setup** de la **Operación de Plataforma**:
   - En caso de **registro de difusión**, el **Anfitrión ECI** presentará el target_id en la pantalla junto con el recuadro de diálogo de registro de la **Operación de Plataforma**. El **Anfitrión ECI** obtendrá la estructura **as_setup** del carrusel de establecimiento de AS (véase la cláusula 7.7.2).

   NOTA 1 – Si una plataforma proporciona varios tipos de **Clientes ECI**, la **Operación de Plataforma** puede solicitar al **Usuario** que ofrezca información adicional a fin de proporcionar el **as_setup** para el tipo de **Cliente ECI** adecuado.

   NOTA 2 – La **Operación de Plataforma** puede asumir que el **CPE** haya descargado la última versión de la **Imagen de Cliente ECI**, y proporcionar la estructura **as_setup** sólo para esa **Imagen de Cliente ECI**.

   - En caso de **registro en línea**, el **CPE** registrará la identificación del cliente, el target_id del **CPE** y la palabra de ocasión utilizando la API web tal como se indica en la cláusula 7.3.3.

   NOTA 3 – La **Operación de Plataforma** puede decidir la aplicación de la palabra de ocasión para asegurar la renovación del registro en cada evento de reinicialización del **Anfitrión ECI**.

Tras la secuencia de adquisición **as_setup** en modo registro, o cuando se ha recuperado la estructura **as_setup** de la memoria NV en modo registrado, el **Anfitrión ECI** inicializará la AS e intentará la carga del **Cliente ECI** encriptado:

1) Carga de la estructura **as_setup** en la AS utilizando el mensaje **reqAsClientImageDecrKey**. Se carga la cadena de clientes certificados **Cliente ECI** en la AS. Se carga la Lista de Revocación de clientes de **Operación de Plataforma** y la cofirma de cliente de **Operación de Plataforma**. Como mínimo, se informarán de forma clara al **Usuario** de los casos de fallo siguientes, o bien se tratarán de forma automática:

   a) Versión antigua del **Anfitrión ECI** – es necesario actualizar el **Anfitrión ECI** o sus credenciales.

   b) Versión antigua del **Cliente ECI** – es necesario actualizar el **Cliente ECI** o sus credenciales.

2) Desencriptación de la imagen utilizando, si es necesario, la clave de la **Imagen** de Cliente calculada para la AS, y autenticación de la imagen de **Cliente ECI** utilizando la firma de **Cliente ECI** y las cofirmas de **Operación de Plataforma**.

3) Fallo en caso de error de validación.

La estructura as_setup y el formato de as_setup_file serán conformes con la definición que figura en el Cuadro 7.8.3-1.

Cuadro 7.8.3-1 – Estructura, fichero y fichero colector de la configuración de AS

Sintaxis	N.° de bits	Mnemónico
ECI_As_Setup {		
as_version	8	uimsbf
if (as_setup_version == 0x01) {		
vendor_id	20	uimsbf
if (/* imagen de cliente ordinario */){		
ECI_Client_id **client_id**		
}		
if (/* series de imágenes de cliente */){		
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**		
}		
}		
}		

Semántica:

vendor_id: entero	Suministrador de seguridad del **Cliente ECI** objeto de este as_setup.
client_id: ECI_Client_Id	ID del **Cliente ECI** objeto de este as_setup. La sentencia 'if' precedente utiliza el client_id del campo tipo: debe corresponder a una "imagen de cliente ordinario".
series_id: ECI_Client_Series_Id	ID de las **Series de Clientes ECI** objeto de este as_setup. La sentencia 'if' precedente utiliza el client_id del campo tipo: debe corresponder a la "series de imagen de cliente".
target_id: ECI_Image_Target_id	ECI_Image_Target_Id que identifica el **CPE** objeto de este mensaje.
as_tag: entero	Etiqueta que indica la versión de la estructura as_setup para el objetivo antes señalado. El valor se modifica por cualquier cambio que se produzca en la estructura de **as_setup** para este objetivo; por ejemplo, un incremento.
online: booleano	Si es verdadero, este mensaje requiere utilizar la palabra de ocasión de intervalo en el mecanismo AK (clave de autenticación); si es falsa, no se requiere palaba de ocasión. Nota – Este bit sólo se podrá a uno en caso de funcionamiento con conexión en línea.
min_root_state: minEciRootState	Estado de raíz mínima (número mínimo de la versión de raíz, número mínimo de Lista de Revocación raíz) aplicable a la validación del **Anfitrión ECI** y los **Clientes ECI** cargados. El campo se codifica como una secuencia de bytes tal como se define en [UIT-T J.1014].
inputV: InputV	Mensaje InputV para el Sistema AS. El campo se codifica como una secuencia de bytes tal como se define en [UIT-T J.1014].
eKey: SymKey	Clave simétrica encriptada para desencriptar la imagen. El campo se codifica como una secuencia de bits, como se define en [UIT-T J.1014].
extension: Extension	Datos de extensión, con retrocompatibilidad. No debe exceder de 256 bytes para las aplicaciones de difusión al objeto de mantener un carrusel de difusión de tamaño reducido. No se define ninguna aplicación para este dato.
magic_file: byte[3]	Representación de la cadena 'AES' con caracteres ASCII de 8 bits.
as_setup_file_version: entero	Versión del formato de ECI_AS_Setup_File. Se reservan los valores 0 y 0x2..0xff. El valor 0x01 se utiliza para el formato aquí definido.
as_setup: ECI_As_Setup	Estructura **as_setup** de la **Operación de Plataforma** para cargar un **Cliente ECI** encriptado específico o un **Anfitrión ECI** específico.
magic_bucket_file: byte[3]	Representación de la cadena 'AEB' con caracteres ASCII de 8 bits.
as_setup_item: ECI_As_Setup	Estructuras as_setup en este colector. Cualquier nueva estructura as_setup se añadirá en la parte superior del colector; por lo tanto, se coloca la estructura as_setup más antigua en el fondo. Las estructuras as_setup sólo se suprimirán, en caso necesario, en el fondo del colector. Los **CPE** pueden así inspeccionar las actualizaciones más rápidamente. Es decir, tras una primera verificación, sólo las estructuras as_setup necesarias se comprobarán de arriba a bajo hasta que se encuentra la primera de las series de verificación anteriores.

La frecuencia mínima de comprobación de las actualizaciones de la estructura **as_setup** será la misma que para otros datos de **Cliente ECI** definida en la cláusula 7.3.1. Nótese que una actualización normalmente implica la actualización del software del **Cliente ECI** y/o del **Anfitrión ECI** del **CPE**; por lo tanto, cualquier actualización de los mismos también se descargará para garantizar que se completa una secuencia de inicialización de **Cliente ECI** coherente. Si ese nuevo conjunto coherente no está disponible, puede utilizarse el conjunto coherente anterior.

Cuando el **Anfitrión ECI** se encuentra intentando completar el registro en modo difusión (manual) de un **Cliente ECI** nuevo o actualizado, el **Anfitrión ECI** comprobará con la máxima frecuencia posible la necesidad de actualización del carrusel de fichero as_setup.

7.8.4 Protocolo de transporte

7.8.4.1 Protocolo de difusión

El protocolo de difusión para estructuras **as_setup** será conforme con la cláusula 7.7.2.

El número de estructuras as_setup que deben actualizarse en un cambio de versión de **Cliente ECI** puede ser muy importante. Para limitar el número de nuevos mensajes as_setup en línea durante un cambio de versión de **Cliente ECI** en una gran operación de sólo difusión, la **Operación de**

Plataforma puede poner a disposición un nuevo **Cliente ECI**, y presentar las nuevas credenciales *por fases*, sustituyendo grupos de **Clientes ECI** en los **CPE**; esto puede repetirse varias veces al objeto de abarcar al mayor número posible de **CPE** antes de utilizar el sistema de seguridad para hacer obligatorio el uso del nuevo **Cliente ECI**.

7.8.4.2 Protocolo en línea

El protocolo en línea es un protocolo sencillo de petición-contestación entre el **CPE** y el **Cliente ECI** que se define en la cláusula 7.7.3, que transfiere como parte de la petición el **target_id** del **CPE** y la **palabra de ocasión** ("**nonce**"), y a lo que se devuelve el **ECI_As_Setup_File**.

7.8.5 Presentación del ID de objetivo al Usuario

El **Anfitrión ECI** y el **Cliente ECI** deben poder presentar el target_id del **CPE** al **Usuario** en redes de difusión cuando no exista una conexión en línea que permita generar información específica del **CPE** para desencriptar la **Imagen de Cliente ECI**, si es necesario, y permitir la generación de mensajes InitV del sistema AS del **Cliente ECI** (el **Cliente ECI** define el protocolo de transporte para esos mensajes). Asimismo, el target_id puede figurar impreso en la carcasa del **CPE** o incluirse en la documentación del mismo. En esta cláusula se define la presentación del target_id al **Usuario**.

El target_id es un entero de 64 bits. Se presentará al **Usuario** según las normas indicadas en la cláusula 6.2.2, utilizando una verificación de suma de 9 bits y añadiendo subcadenas de 9 bits en lugar subcadenas de 5 bits. Por lo tanto, el target_id se representa como una secuencia de seis números de 4 dígitos con dígitos comprendidos entre 0 y 7.

Los **CPE** y los **Clientes ECI** pueden utilizar representaciones a medida en sus interfaces de **Usuario** (por ejemplo, basadas en un esquema de numeración privado del **CPE**) pero siempre ofrecerán funciones de registro de **Cliente ECI** utilizando el formato de presentación antes indicado.

8 Revocación

8.1 Introducción

Todas las partes y los elementos mediante los que estas contribuyen al **Ecosistema ECI** deberán ser certificados por la **TA ECI**. Mediante esa certificación será posible proporcionar una calidad básica adecuada en términos de funcionalidad y de robustez de las implementaciones, así como medidas de renovación pertinentes por las partes que contribuyen. Este proceso de certificación también impide la utilización del ecosistema de la **ECI** mediante operaciones de intrusión y piratería.

La **ECI** proporciona la funcionalidad necesaria para excluir selectivamente la distribución de servicios a los **CPE** en base a la situación de la **TA ECI** del hardware del **CPE**, el **Anfitrión ECI**, otras **Operaciones de Plataforma** y los **Clientes ECI** cargados.

La **TA ECI** puede revocar una **Operación de Plataforma** si esta no sigue normas comúnmente aceptadas, entre otras las relativas a la no interferencia con otras **Operaciones de Plataforma** en **CPE** compartidos, o sobre servicios de piratería distribuidos a través de la **ECI**. Igualmente, la **TA ECI** puede revocar **Clientes ECI** que no respeten las normas generalmente convenidas, entre otras la no interferencia con otros **Clientes ECI** en los **CPE** compartidos o el pirateo informático. La **TA ECI** también puede revocar versiones software del **Anfitrión ECI** si estas presentan aspectos inadecuados que dejan al descubierto secretos de los **Clientes ECI** o que permiten su manipulación.

En todos los casos mencionados, las organizaciones responsables del elemento revocado pueden solventar la deficiencia, normalmente sustituyendo el elemento revocado por uno nuevo. Por lo tanto, un **Suministrador de Seguridad** puede sustituir un **Cliente ECI** con una nueva versión, un **Fabricante de CPE** puede ofrecer parches de seguridad para un **Anfitrión ECI** y un **Operador** puede mejorar sus operaciones con una nueva versión de su **Certificado de Operación de**

Plataforma. Todas estas operaciones se realizan en colaboración y se sugiere ejecutarlas después de haber llegado a acuerdos contractuales entre las partes afectadas y la **TA ECI**.

En caso de que haya participantes en la **ECI** que violen sistemáticamente los acuerdos con la **TA ECI**, afectando negativamente a otras partes o a **Usuarios**, todos sus elementos podrán ser revocados por la **TA ECI**.

Si algunos **CPE** han dejado de tener un **Anfitrión ECI** válido y no se prevé que reciban una actualización de su **Fabricante de CPE**, pueden ser revocados. Esto también ocurre en caso de que el cargador inicial del **CPE** se vea comprometido y permita la descarga de software de **Anfitrión ECI** no conforme.

Los **CPE** intentarán sustituir automáticamente una versión revocada con una versión actualizada que esté disponible. No obstante, pueden bloquearse las nuevas descargas y **Listas de Revocación**. En este caso, una **Operación de Plataforma** puede denegar la prestación de servicios o la presentación de contenidos almacenados en un **CPE** de esas características.

8.2 Revocación de un CPE

La **ECI** permite a las **Operaciones de Plataforma** excluir la provisión de servicios a **CPE** específicos mediante una funcionalidad de oferta selectiva de derechos de un sistema CA o DRM. La **Operación de Plataforma** puede examinar el estado más reciente de la **TA ECI** de un **CPE**. Si la **TA ECI** considera necesario revocar un **CPE**, la **Operación de Plataforma** puede inhabilitar la provisión de servicios al **CPE** en función del ID de su juego de chips registrado con el sistema CA o DRM prestador de servicios.

La presente Recomendación también permite que las **Operaciones de Plataforma** excluyan la prestación de servicios a **CPE** en los que se ejecutan **Anfitriones ECI** revocados. La **Operación de Plataforma** puede utilizar el sistema de Seguridad Avanzada para exigir un número de versión mínimo del **Anfitrión ECI** conforme a una Lista de Revocación de **Anfitrión ECI** reciente, tal como se define en la cláusula 8.3.

Si se considera adecuado, el mecanismo de revocación del **Anfitrión ECI** también puede utilizarse para la revocación de los **CPE**, especificando un número de versión mínimo de **Anfitrión ECI** que sea superior a los generados hasta ese momento.

8.3 Proceso de revocación genérica

En esta cláusula se denomina "versión mínima de **Lista de Revocación**" a la combinación de la versión mínima de **Raíz** y de la versión mínima de la Lista de Revocación **Raíz**.

El mecanismo último de imposición de la revocación de un **Anfitrión ECI** es el agotamiento del servicio: si un elemento revocado está presente en el **Anfitrión ECI** pese a la aplicación de **Listas de Revocación** (presumiblemente anticuadas), la **Operación de Plataforma** puede detener la prestación de servicios a ese **Anfitrión ECI**. El **Sistema AS** protege la distribución de la Lista de Revocación mínima aceptable de una **Operación de Plataforma**: su manipulación causará el agotamiento del servicio. Por lo tanto, una **Operación de Plataforma** puede obligar a que se verifique la versión de las credenciales utilizadas para instalar el **Anfitrión ECI** y todas las demás **Operaciones de Plataforma** y **Clientes ECI**.

La **Operación de Plataforma** proporcionará un servicio de descarga de la **Lista de Revocación** a cualquiera de los elementos anteriores (**Anfitriones ECI**, **Clientes ECI** y **Operaciones de Plataforma**). Ello garantiza la disponibilidad de las listas de revocación más recientes para todos los **Clientes ECI** y **Operaciones de Plataforma** cargadas en el **Anfitrión ECI**.

La inicialización del **Sistema AS** [UIT-T J.1014] permite al **Anfitrión ECI** especificar la versión mínima esperada de la **Lista de Revocación** para todos los elementos. Se utiliza para validar la versión de la Lista de Revocación utilizada de forma retrospectiva por el **Anfitrión ECI**. El **Anfitrión ECI** utilizará el valor mínimo de la **Lista de Revocación Raíz** de los **elementos Cliente ECI** que desea cargar y de la **Imagen de Anfitrión ECI** que ha cargado.

NOTA – Se recomienda que un **Anfitrión ECI** no cargue elementos que ulteriormente causarían una revocación, y que en lugar de ello lo notifiquen al **Usuario**.

Para evitar el agotamiento indebido de un servicio, deben estar disponibles en un **Anfitrión ECI** las credenciales más recientes (y si es necesario las últimas versiones) de todos los elementos a cargar. Para evitar que los Cliente ECI no puedan trabajar adecuadamente por amenazas a la seguridad debidas a la presencia de Certificados de Operación de Plataforma de Anfitrión ECI o Clientes ECI revocados, el **Anfitrión ECI** proporcionará las funcionalidades siguientes a fin de garantizar que las credenciales y (si es necesario) elementos más recientes estén disponibles para evitar el agotamiento indebido del servicio:

- Mantendrá la última cadena de **Lista de Revocaciones de TA ECI** de cada elemento que se verifica en la configuración actual de su **Anfitrión ECI**, **Operación de Plataforma** y **Cliente ECI**, utilizando los servicios de descarga de credenciales y de **Lista de Revocación** del **Fabricante CPE** y de la **Operación de Plataforma** de sus **Clientes ECI**.
- Los ajustes por defecto para todos los modos pertinentes del **CPE** permitirán dicha descarga.
- El **CPE** no tendrá un modo de funcionamiento que impida permanentemente la descarga que no sea la desconexión de la alimentación de energía o la inhibición del acceso a la red de descarga (que no se deba al estado del **CPE** o al modo de operación).
- Se podrán restaurar los ajustes por defecto relativos a la descarga y la revocación por defecto de **Clientes ECI** y de **Operaciones de Plataforma** con una sencilla actuación por parte del **Usuario**.

La presente Recomendación permite a los **Usuarios** anular el comportamiento por defecto del Anfitrión para revocar elementos que causen el agotamiento del servicio de otros. Si algunos **Usuarios** lo hacen (por ejemplo, mantener en ejecución un cliente antiguo) pueden experimentar cada vez más dificultades para la prestación de servicios actualizados.

8.4 Revocación de un Anfitrión ECI basada en Listas de Revocación

Un **CPE** que no tenga un mantenimiento adecuado puede contener un **Anfitrión ECI** revocado. Los **Fabricantes de CPE** deben proporcionar credenciales actualizadas, incluida la última **Lista de Revocación ECI** aplicable. Además, una **Operación de Plataforma** que desee operar un **Cliente ECI** en un **Anfitrión ECI** puede proporcionar un servicio de descarga para una **Lista de Revocación** relativa a las credenciales del **Anfitrión ECI** y, asimismo, puede proporcionar un servicio de descarga para **Anfitriones ECI** seleccionados. El **Anfitrión ECI** aplicará las **Listas de Revocación** a las credenciales del **Anfitrión ECI (Certificado Raíz** y **Certificado de Fabricante)** con arreglo a las normas de procesamiento genéricas de **Listas de Revocación** definidas en [UIT-T J.1014].

El formato del fichero de datos de revocación del **Anfitrión ECI** se define en la cláusula 5.3.

8.5 Revocación de una Operación de Plataforma ECI

Una **Operación de Plataforma** que desee operar un **Cliente ECI** en un **Anfitrión ECI** puede proporcionar un servicio de descarga para una **Lista de Revocación** relativa a otras credenciales de **Operación de Plataforma**. El **Anfitrión ECI** aplicará las **Listas de Revocación** a todas las credenciales de **Operación de Plataforma** instaladas con arreglo a las normas de procesamiento genéricas de **Listas de Revocación** definidas en [UIT-T J.1014].

El formato del fichero de revocación de la **Operación de Plataforma ECI** se define en la cláusula 7.6.3.

8.6 Revocación de un Cliente ECI

Una **Operación de Plataforma** que desee operar un **Cliente ECI** en un **Anfitrión ECI** puede proporcionar un servicio de descarga para una **Lista de Revocación** relativa a otros **Clientes ECI**. El **Anfitrión ECI** aplicará las **Listas de Revocación** a todas las credenciales de **Cliente ECI** instaladas con arreglo a las normas de procesamiento genéricas de **Listas de Revocación** definidas en [UIT-T J.1014].

El formato del fichero de revocación del **Cliente ECI** se define en la cláusula 7.6.3.

9 Interfaces del Cliente ECI

9.1 Introducción

9.1.1 Arquitectura de las interfaces del Cliente ECI

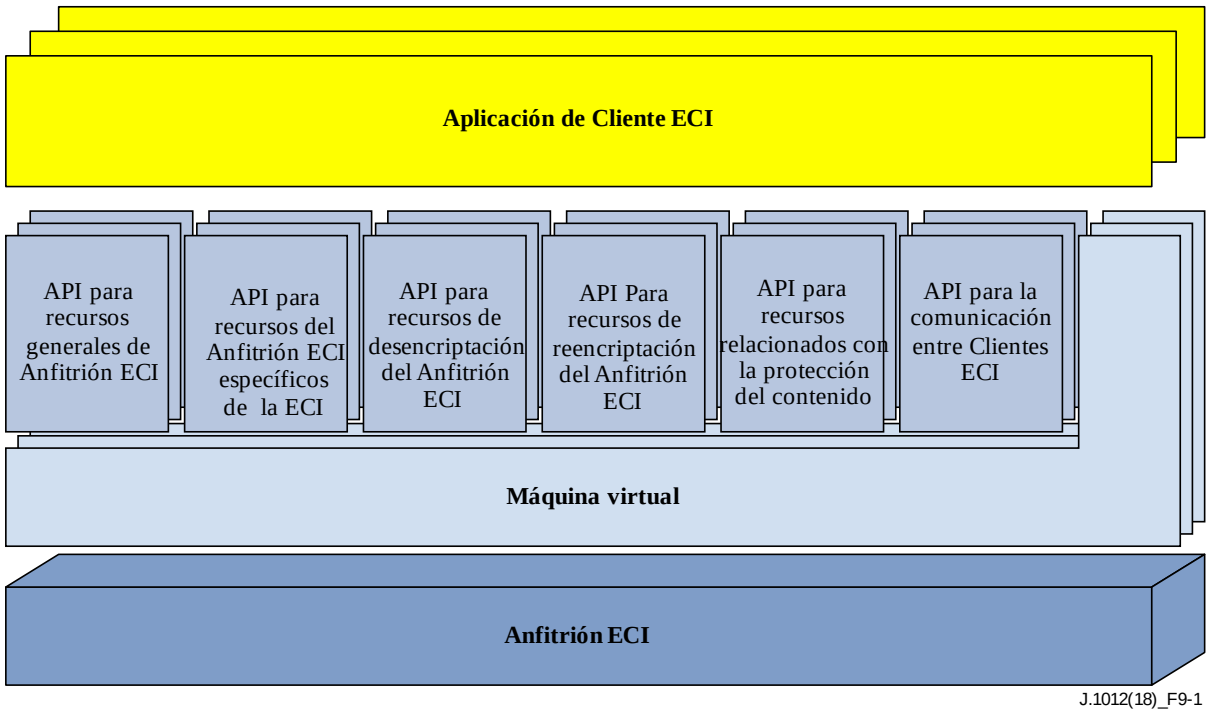


Figura 9.1.1-1 – Estructura de las API definidas en la cláusula 9

La Figura 9.1.1-1 ofrece una visión general de la estructura de las API del sistema **ECI**. Se muestran seis bloques de varias API que puede utilizar el **Cliente ECI**. Esos bloques de API se describen en las cláusulas 9.4 a 9.9. En el Cuadro 9.1.1-1 figuran las API definidas en la cláusula 9 de esta Recomendación; véase también [b-ETSI GS ECI 002].

Cuadro 9.1.1-1 – Lista de las API definidas en esta Recomendación

Cláusula N.°	Categoría de API	Descripción
9.4	API para recursos generales del **Anfitrión ECI**	Son las API que soportan funcionalidades generales del **Cliente ECI**
9.5	API para recursos del **Anfitrión ECI** específicos de la **ECI**	Las API que soportan funcionalidades específicas **ECI** del **Cliente ECI**
9.6	API de acceso a recursos de desencriptación del **Anfitrión ECI**	Las API que permiten al **Cliente ECI** utilizar los recursos de desencriptación del **Anfitrión ECI**
9.7	API de acceso a recursos de re-encriptación del **Anfitrión ECI**	Las API que permiten el **Cliente ECI** utilizar los recursos de re-encriptación del **Anfitrión ECI**
9.8	API para recursos relativos a las propiedades del contenido	Las API que soportan funcionalidades de protección del contenido del **Cliente ECI**
9.9	API para la comunicación entre **Clientes ECI**	Las API que soportan la comunicación directa entre **Clientes ECI**

9.1.2 Asa de Medios

Un **Asa de Medios** es un identificador de un objeto en el entorno del anfitrión que proporciona el contexto para todas las interfaces del **Anfitrión ECI** proporcionadas al **Cliente ECI** en términos de control del proceso de desencriptación de un elemento de contenido. El **Asa de Medios** también permite al **Cliente ECI** especificar los datos que necesita del contenedor de contenidos para poder desaleatorizarlo. En caso de distribución mediante red de difusión también permite controlar la selección del programa a decodificar y la selección del flujo en la red de distribución (función de sintonización). Un **Cliente ECI** también puede solicitar un **Asa de Medios** con acceso a un sintonizador a fin de acceder a datos necesarios para el funcionamiento del **Cliente ECI** en los flujos de red no accesibles por la aplicación/anfitrión con fines de adquisición de contenido. Para la distribución basada en ficheros y flujos OTT, el **Asa de Medios** proporciona una forma de acceso del **Cliente ECI** a datos de seguridad en el fichero/flujo cuya posición en una ubicación normalizada no ha sido especificada.

La desaleatorización de la sesión de medios está directamente bajo el control del **Cliente ECI**. La sincronización de la aplicación de la palabra de control (CW) con el flujo de transporte (TS) se basa en la aleatorización de información de control del TS. La sincronización de CW (que en este contexto se denominan normalmente claves) con un fichero ISOBMFF de CENC [ISO/CEI 23001-7] se basará en identificadores KeyID de CENC.

En el Cuadro 9.1.2-1 figuran sesiones que utilizan un **Asa de Medios**.

Cuadro 9.1.2-1 – Tipos de Asa de Medios

Nombre	Valor	Descripción
MhDvbTs	0x01	TS será conforme con [ISO/CEI 13818-1-1].
MhIsobmffCenc	0x10	El fichero ISO BMFF será conforme con [ISO/CEI 23001-9] e [ISO/CEI 14496-12].
RFU	otros	Reservado para uso futuro.

9.2 Interfaz de la máquina virtual ECI

9.2.1 Principios

Para cada **Cliente ECI** se creará una instancia de máquina virtual diferenciada. En la cláusula 7 se define cómo debe realizarse la carga de datos e instrucciones para un **Cliente ECI** en una máquina virtual (VM).

El funcionamiento de la máquina virtual se define en [UIT-T J.1013]; véase también [b-ETSI GS ECI 001-4].

Todas las interacciones del **Cliente ECI** con el mundo exterior se realizarán utilizando la interfaz de mensajes definida en la cláusula 9.2.3.

9.2.2 Instrucciones y datos (recursos estáticos)

La VM ejecutará las instrucciones que le proporciona el **Cargador de Cliente ECI** como parte del segmento o segmentos de código de la **Imagen de Cliente ECI**.

La VM asegura que las instrucciones no sean automodificables. Cualquier código que conduzca con facilidad a un comportamiento de un **Cliente ECI** no deseable y/o de fácil manipulación (por ejemplo, intérpretes) se considera inadecuado y su exclusión debe garantizarse como parte del proceso de certificación de **Clientes ECI**.

El tamaño máximo del código y el espacio de datos estáticos que necesita un **Cliente ECI** se definen en [b-UIT-T J Supl. 7].

9.2.3 Interacción con el Anfitrión ECI

Todas las interacciones del **Cliente ECI** con el **Anfitrión ECI** se definen en base al modelo de mensajes de esta cláusula. El **Cliente ECI** y el **Anfitrión ECI** sólo comparten los datos siguientes:

- datos contenidos en mensajes;
- cualquier dato almacenado en memoria NV del **Anfitrión ECI** en nombre del **Cliente ECI**; o
- cualquier dato presente en los canales de comunicación hacia o desde otros **Clientes ECI**.

Obsérvese que estos datos también se intercambian a través de mensajes.

El modelo del mensaje se basa en tres tipos distintos de intercambios desde **Cliente ECI** al **Anfitrión ECI**:

1) Un **Cliente síncrono** ha iniciado el intercambio: el **Cliente ECI** llama a una **Función de Anfitrión ECI** que reacciona en un tiempo muy breve. El hilo (flujo de ejecución) del **Cliente ECI** queda bloqueado mientras el **Anfitrión ECI** procesa el mensaje y envía un mensaje de vuelta.

2) Un **Cliente asíncrono** ha iniciado el intercambio: el **Cliente ECI** envía al **Anfitrión ECI** un mensaje **Petición de Cliente** que se pondrá en cola y será procesado por el **Anfitrión ECI** a su debido tiempo. La llamada asíncrona generará un mensaje de **Retorno** inmediato con un único resultado básico (identificador de mensaje o error). El **Anfitrión ECI** proporcionará posteriormente una **Contestación de Anfitrión** en la que informará de la situación y resultados de la operación del **Anfitrión ECI** iniciada por el **Cliente ECI**.

3) Un **Anfitrión asíncrono** ha iniciado el intercambio: el **Anfitrión ECI** transmite al **Cliente ECI** un mensaje que se pondrá en cola y será procesado por el **Cliente ECI** a su debido tiempo. La llamada asíncrona generará un mensaje de **Retorno** inmediato con un único resultado básico (normalizado). El tipo y formato de este mensaje, tal como se representa en el **Anfitrión ECI**, está fuera del alcance de la presente Recomendación, ya que es un asunto interno del **Anfitrión ECI**.

   Obsérvese que sólo se define la representación para el **Cliente ECI**. El **Cliente ECI** proporcionará más tarde un mensaje de **Contestación** con la situación y resultados de la operación del **Cliente ECI** iniciada por el **Anfitrión ECI**.

Los distintos tipos de intercambios de mensajes entre el **Anfitrión ECI** y el **Cliente ECI** se muestran en la Figura 9.2.3-1.

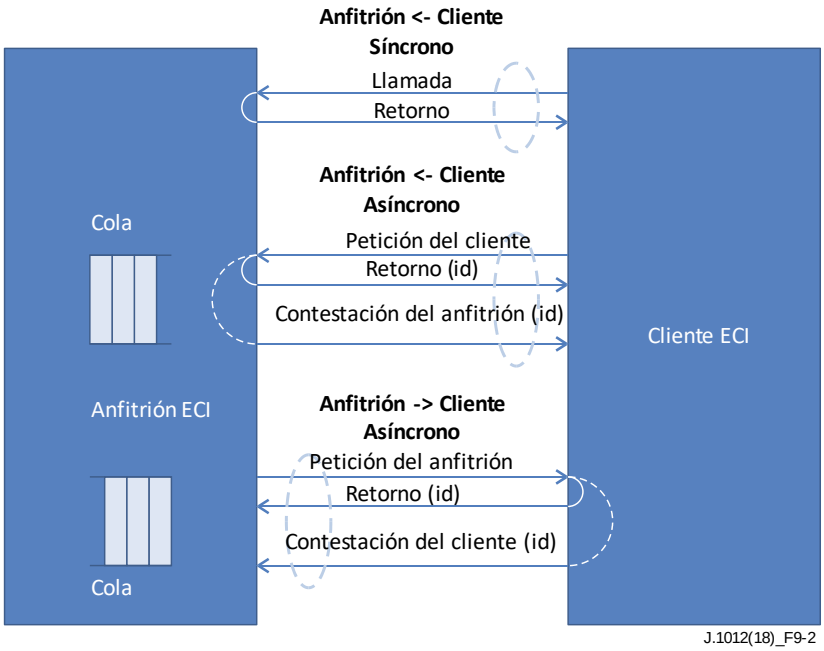


Figura 9.2.3-1 – Intercambios de mensajes entre Cliente y Anfitrión

El **Cliente ECI** debe asegurar que la carga útil esté protegida según sea necesario, por ejemplo, con palabras de control y propiedades del contenido. Además, la interfaz no está diseñada para realizar intercambios de contenidos ni estos constituyen su objeto.

El **Cliente ECI** implementará Contestaciones a las **Peticiones del Anfitrión ECI** que soporta, con arreglo a las definiciones de API tal de la cláusula 9 utilizando el identificador de cada **Petición** en la respectiva **Contestación**.

El **Anfitrión ECI** implementará Contestaciones a las **Peticiones del Cliente ECI** que soporta, con arreglo a las definiciones de API de la cláusula 9 utilizando el identificador de cada **Petición** en la respectiva **Contestación**.

Una **Petición** asíncrona puede opcionalmente indicar que no es necesaria una **Contestación**. Por ejemplo, cuando se desplazan numerosos elementos de datos y el iniciador sólo requiere **Contestación** a la última **Petición**, asumiendo que todos los elementos de datos intermedios se procesan correctamente.

Todas las **Peticiones de Anfitrión ECI** y las **Contestaciones de Anfitrión ECI** asíncronas se ponen en la cola "en orden de ocurrencia".

9.2.4 Recursos dinámicos proporcionados para un Cliente ECI

Los parámetros técnicos de los recursos dinámicos mínimos que requiere un **Cliente ECI** se especifican en [b-UIT-T J Supl. 7]. Se incluyen los elementos siguientes: hilos, espacio en la pila, espacio de acumulación, tiempo de ejecución, almacenamiento NV y comunicación entre clientes.

9.2.5 Gestión de la versión de API

Las API definidas en la presente Recomendación pueden tener varias versiones, por ejemplo, para ofrecer una funcionalidad mejorada que sustituya a una funcionalidad anterior o para corregir deficiencias de la especificación. Durante la inicialización, los **Clientes ECI** y sus **Anfitriones ECI** necesitan determinar cuáles son las API que soportan sus pares y seleccionar la versión de cada API disponible que se utilizará durante el resto del ciclo de vida del **Cliente ECI**. Los **Clientes ECI** no pueden utilizar API distintas a las descubiertas en el proceso de inicialización ya que las versiones de los mensajes (es decir, su disponibilidad, longitud y sintaxis) no se definen hasta que se completa el proceso de descubrimiento.

Las versiones de las API son autocontenidas en cuanto a su semántica, es decir, la interacción de mensajes entre **Cliente ECI** y **Anfitrión ECI** a través de una versión de la API no depende de que se soporten otras versiones de esa API en el **Anfitrión ECI** ni de las interacciones del **Anfitrión ECI** con otros **Clientes ECI** utilizando otras versiones de esa API.

NOTA 1 – Por motivos prácticos, el texto de las cláusulas que definen nuevas versiones de API puede hacer referencia a textos que definen versiones anteriores de la API en esta Recomendación.

Las API pueden ser obligatorias, opcionales o condicionadas (obligatorias sujetas a una condición). Un ejemplo de condicionalidad es que la API relativa al PVR sea soportada por un **CPE** que, a su vez, soporte el PVR. Futuras versiones de la presente Recomendación podrán definir perfiles de API que sean soportadas por **Anfitriones ECI** y **Clientes ECI** que hagan referencia al nombre de perfil o al número de versión de la especificación.

Para la conformidad con la presente Recomendación y a fin de garantizar la retrocompatibilidad, un **Anfitrión ECI** o un **Cliente ECI** que soporte una API soportará todas las versiones de esa API (incluida la última disponible) salvo hayan sido explícitamente descartadas en (futuras versiones de) la presente Recomendación o se haya declarado expresamente algo diferente.

NOTA 2 – Una versión futura de la presente Recomendación no implica que los **Clientes ECI** o los **Anfitriones ECI** instalados o nuevos deban ser conformes con la misma. Cualquier política de actualización de la planta de **Anfitriones ECI** o **Clientes ECI** con arreglo a nuevas normas o versiones de especificación que obliguen a disponer de nuevas versiones aplicables a nuevos **Anfitriones ECI** y **Clientes ECI** queda fuera del alcance de la presente Recomendación.

Los **Clientes ECI** deben seleccionar el número de versión más alto de las API disponibles en los **Anfitriones ECI** que pueden gestionar y, a la inversa, los **Anfitriones ECI** seleccionan el número de versión más alto de una API en los **Clientes ECI** que gestionan. Ello alienta la migración a versiones superiores más maduras de las API y evita problemas asociados a versiones antiguas en caso de que se descarten versiones (más antiguas) de la API.

A la vista del ciclo de vida típicamente más largo de los **Anfitriones ECI** y la relativa facilidad para la actualización de los **Clientes ECI**, estos deben soportar versiones más antiguas de una API de un **Anfitrión ECI** que refleje la planta instalada (que puede ser objeto de acuerdos adicionales fuera del alcance de la presente Recomendación). A la inversa, los nuevos **Anfitriones ECI** deben soportar **Clientes ECI** más antiguos, reflejo de la planta de **Clientes ECI** (que pueden estar sujetos a acuerdos posteriores, fuera del alcance de la presente Recomendación).

La API para el descubrimiento de la relación **Cliente ECI**-**Anfitrión ECI** se define en la cláusula 9.4.2.

9.2.6 Supervisión de la capacidad de respuesta

El **Anfitrión ECI** ejecutará algunas funciones básicas automáticas de reinicio del **Cliente ECI** en aras de una robustez adicional a la funcionalidad general del **CPE**. El **Anfitrión ECI** detectará situaciones de error fatal en el **Cliente ECI** y en ese caso lo reinicializará automáticamente. Todos los recursos utilizados por el **Cliente ECI** serán liberados antes de la reinicialización, incluyendo las **Asas de Medios,** sesiones mmi, ficheros, conexiones IP, etc.

Se definen las siguientes condiciones de error:

- El **Anfitrión ECI** supervisará la ejecución de cualquier instrucción no permitida por el código del **Cliente ECI**, como el código operacional (opcódigo) de instrucciones no definidas, el direccionamiento de datos no permitido o el direccionamiento de código inexistente, el desbordamiento o la infrautilización de la pila de registro, etc.
- El **Anfitrión ECI** utilizará una temporización para la aceptación de un nuevo mensaje por el **Cliente ECI**. Un valor propuesto para este parámetro se indica en [b-UIT-T J Supl. 7].

En caso de reinicializaciones repetitivas, el **Anfitrión ECI** puede aplicar una política que posiblemente conlleve ajustes del **Usuario** o datos de entrada del **Usuario**, a fin de decodificar la exclusión de un **Cliente ECI** con fallos reiterados de forma más permanente.

NOTA – Cualquier ejecución de una llamada del sistema sys_exit (véase [UIT-T J.1013]) por un **Cliente ECI** se interpretará como la terminación ordinaria de un **Cliente ECI**. Ello normalmente implica la posible supresión o sustitución del **Cliente ECI** por una versión posterior. El **Anfitrión ECI** no suprime automáticamente el **Cliente ECI** sobre la base de ese evento, sino que espera hasta la invocación de un procedimiento pertinente de sustitución o supresión a través de otras políticas de gestión de **Clientes ECI**.

9.3 Mecanismos aplicables a las API del Cliente ECI

9.3.1 Sintaxis de un mensaje asíncrono

Todas las estructuras de mensajes se definen en función de su aparición en la VM de la **ECI**. En el Cuadro 9.3-1 se presenta la estructura de la memoria intermedia de mensajes para todos los mensajes asíncronos según su aparición en el mapa de memoria VM. Obsérvese que todas las memorias de mensajes se estructuran se alinean en bloques de 32 bits.

Cuadro 9.3-1 – Sintaxis de los mensajes asíncronos

Sintaxis con notación C	N.° de bits
struct messageBuffer {	
uint32 **msgTag**;	32
uint16 **msgId**	16
uint16 **payloadLen;**	16
uint32 **payload[];**	n*32
} MessageBuffer;	

msgTag:

Este campo representa los valores siguientes:

- Bits 0-15: **msgApiTag**. Identificación de la API correspondiente al mensaje (véase la definición en el Anexo C).
- Bits 16-23: **msgCallTag**. Identificación de llamada de la API, que debe interpretar el receptor en el contexto del valor de **msgTag** y la versión de la API acordada.
- Bits 24-31: **msgFlags**: Banderas adicionales para la cualificación de un mensaje. La definición es la siguiente:
  - Bit 24: **msgNoResFlag**: para los mensajes de **Petición** e invocación: si es 0b1 no se necesita **Contestación** o respuesta; si es 0b0 se necesita una **Contestación** o respuesta. Este bit no tiene significado en mensajes de respuesta y de contestación.
  - Los bits 25-31 están reservados para uso futuro; el iniciador del mensaje los pondrá a 0b0.

La etiqueta del mensaje será la misma para la **Contestación** a mensajes de **Petición** y para respuestas a mensajes de invocación.

msgId (identificador de mensaje):

- Valor del identificador de mensaje del mensaje asignado por el **Anfitrión ECI**. Para un mensaje de contestación corresponderá al valor del mensaje de petición original. El **Cliente ECI** que envía una petición puede dejar sin inicializar este campo (el **Anfitrión ECI** asignará el valor y los devolverá como un valor resultado de la llamada del sistema SYS_PUTMSG.).

payloadLen (longitud de la carga útil):

- El campo longitud de la carga útil representa el tamaño de la memoria intermedia de la carga útil en bytes. El tamaño realmente asignado del campo **carga útil** será ese valor redondeado al múltiplo de 4 más próximo o superior. Al interpretar el campo **carga útil** de un mensaje recibido, los **Anfitriones ECI** verificarán que los datos no sobrepasan lo indicado por **payloadLen**; de no ser así, se devolverá un error. Los **Clientes ECI** pueden asumir que los **Anfitriones ECI** proporcionan memorias intermedias para mensajes correctamente dimensionadas.

payload field (campo carga útil):

- El campo carga útil se utiliza para transportar parámetros del mensaje. La estructura de la carga útil se define mediante sintaxis en notación C de la firma de la llamada a una función utilizando reglas de correspondencia específicas definidas en la cláusula 9.3.2.3.

9.3.2 Convenio para la definición de la estructura de mensajes asíncronos

9.3.2.1 Sintaxis de las definiciones de mensajes

Los mensajes asíncronos se definen utilizando una declaración de firma de función en notación C. Esta notación corresponde a la estructura de los mensajes mediante las reglas definidas en esta cláusula. A continuación figura un ejemplo de declaración de firma de función:

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

9.3.2.2 Tipos básicos de parámetros de mensajes

La sintaxis utilizará los tipos básicos para las definiciones de parámetros tal como se especifica en el Cuadro 9.3.2.2-1.

Cuadro 9.3.2.2-1 – Tipos básicos utilizados para las definiciones de parámetros de mensajes

Tipos básicos	Representa
uint8, uchar, byte:	Entero sin signo de 8 bits
int8, char, bool:	Entero con signo de 8 bits
uint16, ushort:	Entero sin signo de 16 bits
int16, short:	Entero con signo de 16 bits
uint32, uint:	Entero sin signo de 32 bits
int32, int:	Entero con signo de 32 bits
uint64, ulong:	Entero sin signo de 64 bits
int64, long:	Entero con signo de 64 bits
char *, … ,long * (memoria del cliente)	32 bits; solo permitido para mensajes síncronos

Para los parámetros booleanos se utilizan los valores simbólicos **Verdadero** y **Falso**. Según la definición en lenguaje C, **Falso** se representa por 0x00 y **Verdadero** por cualquier valor distinto de 0x00.

9.3.2.3 Correspondencia entre la carga útil del mensaje y los parámetros del mensaje

El campo **carga útil** contiene todos los parámetros del mensaje. El parámetro identificador de mensaje **msgId** y los parámetros de resultados **msgResult** son implícitos en el sentido de que no se incluyen expresamente en la descripción de la sintaxis declarativa de firma de función. Su presencia está implícitamente definida por el tipo de mensaje.

El **Anfitrión ECI** asociará un **msgId** a los mensajes de **Anfitrión ECI** y de **Petición de Cliente ECI** a fin de relacionar la **Petición** con la correspondiente respuesta. El tipo de msgId es uint32. El **Anfitrión ECI** tiene la responsabilidad de gestionar los valores de msgId. Los valores de msgId no se volverán a enviar hasta que se transfiera el mensaje **Contestación**.

La **Contestación** contendrá un parámetro **msgResult** de tipo int32.

Estos parámetros implícitos son los que ocupan las primeras posiciones en el campo carga útil de la memoria intermedia de mensajes. En el Cuadro 9.3.2.3-1 se presenta la secuencia de parámetros del campo carga útil para cada tipo de mensaje desde la perspectiva del **Cliente ECI** (la perspectiva del **Anfitrión ECI** está fuera del alcance en la **ECI**).

Cuadro 9.3.2.3-1 – Tipos de mensajes y parámetros "ocultos" (perspectiva de Cliente)

Tipo de mensaje	Parámetros implicados	Campo carga útil
Petición de cliente, C→H	*Ninguno*	p_1, .. , p_n
Contestación de anfitrión, H→C	msgId, result	msgId, result, p_1, .. , p_n
Petición de anfitrión, H→C	msgId	msgId, p_1, .. , p_n
Contestación de cliente, C→H	msgId, result	msgId, result, p_1, .. , p_n

Las reglas siguientes se utilizarán para adaptar los parámetros (ya sean estructuras, bytes o matrices cortas, etc.) a la estructura de la carga útil de la memoria intermedia de mensajes en el espacio de memoria de **Cliente ECI**:

- Los parámetros se integran en la memoria colocando en primer lugar la parte menos significativa de la dirección, a excepción de los campos de datos de matrices de longitud variable.
- Cualquier tipo de datos de 8 o 16 bits se amplía a 32 bits utilizando la extensión adecuada a su tipo (con signo o sin signo).
- Estructuras (sin incluir campos de bits): todos los campos se organizan respetando el orden en que han sido definidos, alineados según el tamaño del campo (para entidades de 16 y 32 bits) con la dirección más baja en el primer campo y un campo de relleno precediendo al campo de mayor tamaño posterior. La estructura siempre se ajusta con relleno hasta completar del siguiente bloque de 32 bits. Las estructuras de unión se rellenarán hasta completar el mayor tamaño de las alternativas.
- Matrices de bytes (8 bits), cortas (16 bits) e int (32-bit): se incluirán en la memoria intermedia de mensajes (no como punteros a la memoria del **Cliente ECI**). Las matrices de longitud fija utilizarán la notación siguiente: <type>, <array_identifier>, '[' <constant> ']'. Se organizarán en el mismo orden en que aparecen en la lista de parámetros. Las matrices de longitud variable utilizarán la notación <type>, <array_identifier>, '[' ']'. Todas las matrices de longitud variable se organizarán en dos campos de 32 bits. El primer campo incluye el desplazamiento en la memoria intermedia de mensajes donde la que se ubica el primer elemento de la matriz. El segundo campo contiene la longitud de la matriz (en bytes).
- Las entidades de 64 bits se almacenarán con los 32 bits más significativos en primer lugar (según el convenio típico de representación de entidades de 64 bits en máquinas de 32 bits con configuración "little endian").
- Todas las entidades de 32 y 16 bits se representarán en memoria ("endianness") de manera natural (a priori desconocida y que viene determinada por la arquitectura subyacente de la CPU).

- Cualquier (char*) que apunte a caracteres imprimibles utilizará la representación UTF-8 [ISO/CEI 21320] para los "puntos de código" reales salvo que explícitamente se defina otra cosa. Los caracteres pueden representarse con un número de bytes de 1 a 4 (en función del punto de código). En esta especificación no se definen los puntos de código que serán imprimibles en un **CPE** (que pueden tener distintas implementaciones en diferentes regiones).

NOTE – El **Anfitrión ECI** es responsable de interpretar la etiqueta del mensaje junto con la versión de la API acordada con el **Cliente ECI** durante el descubrimiento. Igualmente, el **Cliente ECI** es responsable de interpretar la etiqueta del mensaje conjuntamente con la versión de la API acordada con el **Anfitrión ECI** durante el descubrimiento.

9.3.2.4 Convenio de denominación aplicable a mensajes asíncronos

Convenio aplicable a nombres de función:

Todos los nombres de función comenzarán con una indicación de tres letras que refleja el tipo de mensaje. El <name> (nombre) de la función comenzará con letra mayúscula. A continuación se define el convenio de nombres de los mensajes según su tipo:

`req<name>(): request message; res<name>(): response message;`

EJEMPLO 1: reqIpTcpSend().

Convenio de notación aplicable a parejas de mensajes:

Los mensajes **Petición** y **Contestación** se definen como una pareja, así como los mensajes de invocación y de respuesta. La notación siguiente se utiliza para hacer referencia a dichas parejas de mensajes:

`<requestMessage>` → `<responseMessage>`

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

En aras de la brevedad, las firmas de funciones pueden aparecer en esta y otras notaciones sin incluir los parámetros.

El Cuadro 9.3.2.4-1 proporciona algunos ejemplos prácticos de correspondencias reales de nombres de mensajes con posibles funciones en C utilizando procedimientos de programación de eventos de subscripción/retrollamada en notación javascript basada en procedimientos o bien bucles de despacho. La función **subscr** permite la llamada a una función cuando se recibe un mensaje con etiqueta. Se presentan dos ejemplos: uno es selectivo en función del identificador **msgId** e incluye una estructura **cntxt** en la función. El segundo ejemplo no filtra en base al **msgId** y no incluye una estructura **cntxt** en la retrollamada/despacho.

Cuadro 9.3.2.4-1 – Parámetros del campo carga útil por tipo de mensaje con parámetros p_1, .. ,p_n

Mensaje	Notación de procedimiento	Subscripción de evento de retrollamada de cliente	Notación o invocación de retrollamada/ despacho de cliente
Petición (Req), C→H	id = reqName([tag],p_1,..p_n)		
Contestación (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)
Petición (Req), H→C	[tag =] reqName([id],p_1,..p_n)	subscr(tag,invName)	invName(id,p_1,..p_n)
Contestación (Res), C→H	resName([tag],id,res,p_1..p_n)		

9.3.3 Mensajes síncronos

Los mensajes síncronos adoptan el mismo convenio de notación utilizando nombres de funciones como mensajes asíncronos. Los parámetros de los mensajes síncronos no se configurarán en serie para encajarlos en memorias intermedias de mensajes sino que utilizarán convenios generales establecidos en C para las llamadas a funciones así como la definición de la interfaz binaria de la aplicación de VM a fin de adaptar los procedimientos a la memoria VM y al estado de registro. Ello permite establecer una correspondencia directa entre mensajes síncronos y funciones C ordinarias como parte de la biblioteca de un **Cliente ECI**.

Existen tres tipos predefinidos: **get** (obtener) para leer una variable en el dominio del **Anfitrión ECI**; **set** (fijar) para escribir una variable en el dominio del **Anfitrión ECI** y una función de propósito general **call** (llamada) que devuelve un código de error negativo o un valor de función no negativo tal como se muestra en el Cuadro 9.3.3-1.

Cuadro 9.3.3-1 – Tipos de funciones síncronas

Tipo	Aplicables a	Notación	Resultado	Semántica
Get	Variable de Anfitrión	getVariable((**i1**..**in**)	tipo variable	Leer una variable indexada según los parámetros **i1**..**in** en el dominio del **Anfitrión ECI** (para este **Cliente ECI**) (véase la nota).
Set	Variable de Anfitrión	setVariable((**i1**..**in,** value)	void	Asignar un valor a una variable indexada según los parámetros **i1**..**in** en el dominio del **Anfitrión ECI** (para este **Cliente ECI**) (véase la nota).
Call	Anfitrión	callFunc(p1..pn)	int o void	Realizar una llamada síncrona (de propósito general) a una función en el dominio del **Anfitrión ECI**. El valor devuelto es del mismo tipo que el valor resultado de mensajes asíncronos; es decir, los valores negativos representan la ocurrencia de un error. Algunas funciones pueden ser de tipo void, que no permiten ningún error de señalización.
NOTA – Puede hacerse que el **Anfitrión ECI** adopte medidas adicionales a la devolución del objeto solicitado como consecuencia de la invocación de una función Get.				

Ejemplos de definición de mensajes síncronos:

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

Ejemplos de uso:

```
   uint clock = getClock() ;       /* lectura del reloj */
   setPwrWakeup (1000);               /* fija temporizador del despertador  ;
activa las invocaciones */
   (void) memcpy(ptr1,ptr2,100*1000)  /* copia eficientemente la memoria del
cliente */
```

9.3.4 Códigos de error en mensajes de retorno

Los parámetros del código de Retorno de **Contestaciones**, **Respuestas** y (si procede) **Llamadas** incluirán un entero con signo de 32 bit. Si el valor devuelto es cero o positivo significa que la ejecución del código ha sido satisfactoria. En caso de error, devuelve un valor negativo. Los errores son genéricos (véase el Cuadro 9.3.4-1) o específicos de la **Petición** (véanse los códigos de error específicos de cada **Petición**).

Cuadro 9.3.4-1 – Códigos de error en los mensajes de retorno

Nombre/Constante	Valor	Descripción
	1..MaxInt	**Petición** exitosa, valor definido por las definiciones de mensajes.
ErrReqOkNoId	0	**Petición** exitosa.
ErrReqApiErr	-1	No se soporta la API designada por msgApiTag.
ErrReqCallErr	-2	No se soporta la llamada en la API designada por msgApiTag.
ReqQueueErr	-3	Existe un problema en el encolamiento del mensaje, hay desbordamiento de la cola de la memoria intermedia de la **ECI**.
ReqResource	-4	Se ha producido un problema de recursos durante el procesamiento de la **Petición** (por ejemplo, problema de memoria por un número excesivo de mensajería).
RFU	-5..-15	Reservado para uso futuro (tipos de error genéricos).
ReqParam<N>Err	-16..-48	Error en el parámetro **N** = -Resultado-15.
Reservado para errores de la VM	-49..-64	Los códigos de error están reservados a errores específicos de la VM tal como se define en [UIT-T J.1013].
RFU	-65 .. -256	Reservado para uso futuro.
Error específico de la API	-256 .. -511	Error específico de la API definido en el cuadro de Códigos de error de la API.
RFU	-512.. MinInt	Reservado para uso futuro.

NOTA – Normalmente, un **Cliente ECI** puede apoyarse en el **Anfitrión ECI** para permitir un perfil específico de las API tal como se define en la cláusula 9.2.5, pudiendo gestionarse libremente las memorias intermedias de encolamiento de mensajes. Por lo tanto, normalmente es innecesario un procesamiento inteligente de los errores; habitualmente el código de error sólo se utiliza en casos de depuración del **Cliente ECI**.

Los códigos de error específicos de la API o el ReqParamNErr no pueden devolverse formando parte de un mensaje de Retorno, sino que ese error se señalizará como parte de una **Contestación**.

9.3.5 Canal autenticado seguro

Las API de Seguridad avanzada disponen de herramientas para el establecimiento de un **Canal autenticado seguro (SAC)** entre el **Cliente ECI** y cualquier otro dispositivo pertinente (véase 9.5.2). Si un **Cliente ECI** necesita una comunicación autenticada segura con otro **Cliente ECI** o cualquier dispositivo externo, debe definir un mecanismo propio que puedan utilizar las API disponibles, especialmente las API de Seguridad avanzada.

9.3.6 Verificación del mensaje por el Anfitrión ECI

A fin de evitar que se produzcan errores o la adopción de medidas inadecuadas como consecuencia de **Peticiones** o **Contestaciones** inadecuadas, los **Anfitriones ECI** realizarán una comprobación completa de cualquier mensaje procedente de un **Cliente ECI**. Se realizarán las comprobaciones siguientes:

- Soporte del **msgApiTag**.
- Soporte del **msgCallId** en el espacio de mensajes de la API (en el contexto de la versión de la API establecida en el descubrimiento).
- Verificación de si las limitaciones de la carga útil y específicamente msgLength cumplen las normas de sintaxis del mensaje y que la memoria intermedia de mensajes (para mensajes asíncronos) y si cualquier memoria del espacio de direccionamiento del **Cliente ECI** que el **Anfitrión ECI** debe leer o escribir está limitada a partes definidas del espacio de direccionamiento del **Cliente ECI**.
- Verificación del eventual incumplimiento de alguna **Precondición** específica del mensaje (en el sentido de que la **Precondición** sea esencial para la integridad de la **Petición** o la **Contestación**).
- Verificación de si algún puntero o memoria implicados en el mensaje forma parte de la memoria asignada al **Cliente ECI**.

9.3.7 Procesamiento de mensajes por Clientes ECI

Cualquier memoria asignada para la transmisión de una **Petición** puede ser reutilizada tras el mensaje de retorno, salvo que expresamente se indique lo contrario (normalmente mensajes largos para los que es importante evitar que se produzcan copias). Igualmente, cualquier memoria asignada para el envío de una **Contestación** puede ser reutilizada inmediatamente después del evento de transmisión.

Los **Clientes ECI** no se apoyarán en **Anfitriones ECI** para remitir una **Contestación** a cada **Petición**.

Los **Clientes ECI** pueden verificar la corrección de la sintaxis de cualquier **Petición de Anfitrión ECI** o **Contestación**. Si el formato de la **Petición** o la **Contestación** es incorrecto, el **Cliente ECI** no está obligado a responder para facilitar información al **Anfitrión ECI**.

9.4 Conjunto de las API de recursos generales del Anfitrión ECI

9.4.1 Lista de las API definidas en 9.4

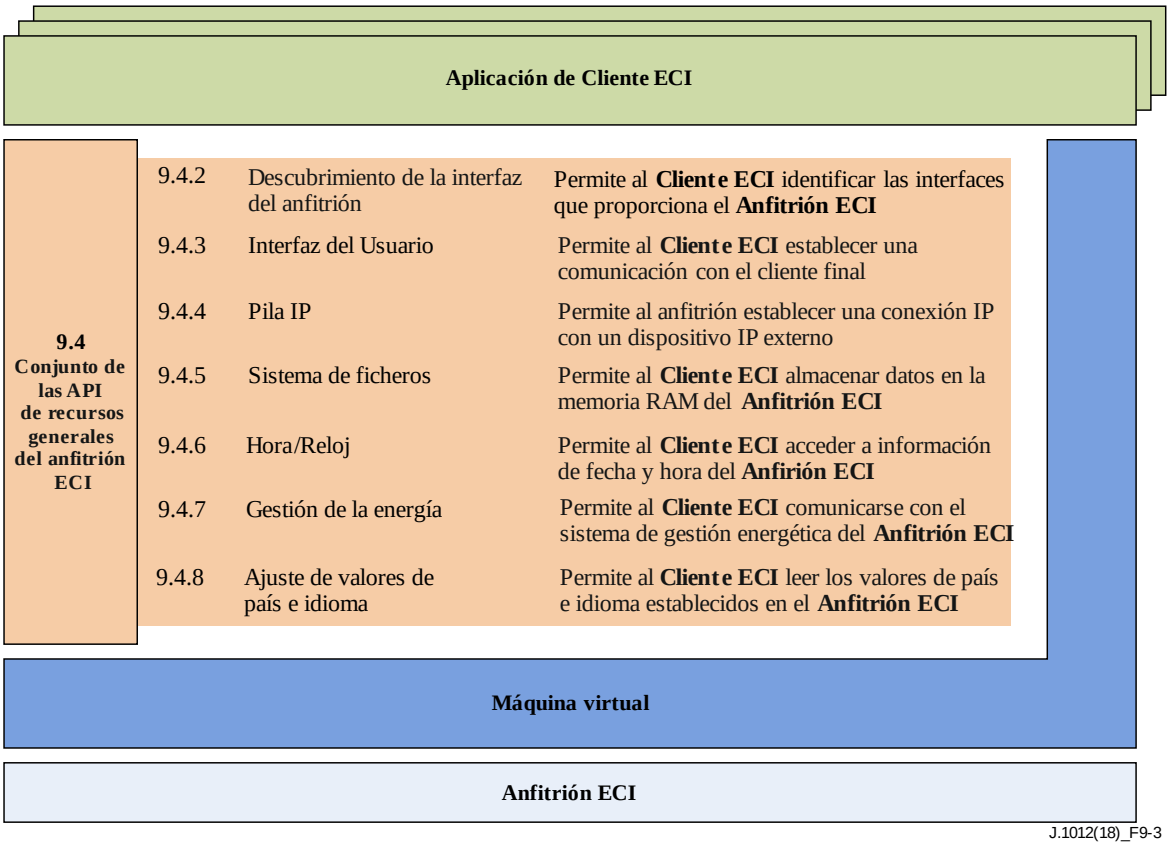


Figura 9.4.1-1 – Representación esquemática de las API definidas en 9.4

Cuadro 9.4.1-1 – Lista de las API definidas en 9.4

Cláusula	Nombre de la API	Descripción
9.4.2	Descubrimiento de la interfaz del anfitrión	Permite al **Cliente ECI** identificar las interfaces que proporciona el **Anfitrión ECI**
9.4.3	Interfaz de Usuario	Permite al **Cliente ECI** establecer la comunicación con el **Usuario**
9.4.4	Pila IP	Permite al anfitrión establecer un enlace IP con un dispositivo IP externo
9.4.5	Sistema de ficheros	Permite al **Cliente ECI** almacenar datos en la memoria RAM del **Anfitrión ECI**
9.4.6	Hora/Reloj	Permite al **Cliente ECI** accede a información de fecha y hora del **Anfitrión ECI**
9.4.7	Gestión de la energía	Permite al **Cliente ECI** ponerse en comunicación con el sistema de gestión energética del **Anfitrión ECI**.
9.4.8	Ajuste de valores de país e idioma	Permite al **Cliente ECI** leer los valores de país e idioma establecidos en el **Anfitrión ECI**

En el Cuadro 9.4.1-1 se muestran las API definidas en la cláusula 9.4 y la Figura 9.4.1-1, lo cual ilustra la ubicación de las API definidas en 9.4 con la **arquitectura ECI**.

En el Cuadro 9.4.1-2 se muestra la estructura del cuadro utilizado para dar una visión general de los Mensajes de presentación relacionados con las distintas API para cada una de ellas.

Cuadro 9.4.1-2 – Estructura del cuadro que resume las funciones de los mensajes API individuales

Mensaje	Tipo	Dirección	Etiqueta	Descripción
Nombre del Mensaje	Véase el Cuadro 9.4.1-3	C→H (cliente→ anfitrión) o H→C (anfitrión → cliente)	Valor de la etiqueta	Breve descripción de la función del Mensaje

La columna Tipo del Cuadro 9.4.1-2 indica el tipo de Mensaje conexo, que puede ser síncrono o asíncrono. En el Cuadro 9.4.1-3 se presenta información adicional. En el Apéndice I se incluye una lista completa de todos los mensajes API disponibles para un **Cliente ECI**.

Cuadro 9.4.1-3 – Posibles valores de la columna Tipo

Categoría del Mensaje	Notación en la columna Tipo	Comentario
Mensaje asíncrono	A	Posibles tipos de mensajes: véase el Cuadro 9.3.2.3-1
Mensaje síncrono	A Set Get Call	Posibles tipos de mensajes: véase el Cuadro 9.3.3-1

9.4.2 API de acceso a los recursos de descubrimiento de la interfaz del Anfitrión ECI

9.4.2.1 Introducción

En esta cláusula se define la API que puede utilizar un **Cliente ECI** para el descubrimiento de las API y de las versiones de las API que soporta el **Anfitrión ECI** y seleccionar la versión más adecuada para la sesión del **Cliente ECI** con el **Anfitrión ECI**. El mecanismo de gestión de la versión de la API permite una selección individualizada de la API. Una vez seleccionada la versión de una API, esta seguirá utilizándose hasta el siguiente evento de inicialización del **Cliente ECI** con el **Anfitrión ECI**.

Las políticas relativas a la disponibilidad de las API se analizan en la cláusula 9.2.5. Las API que son obligatorias se definen en la cláusula 10.

El **Cliente ECI** realizará la gestión de la versión tan pronto como sea inicializado: ninguna API puede utilizarse sin que exista una versión (mutuamente) establecida.

La versión de una API se representará mediante un número de 16 bits. La numeración de las versiones de API comienzan por 0x0000. La asignación ordinaria de nuevas versiones es incremental (de 1 en 1).

En el Cuadro 9.4.2.1-1 figuran los mensajes de la API.

Cuadro 9.4.2.1-1 – API de descubrimiento en la interfaz del Anfitrión ECI

Mensaje	Tipo	Dir.	Etiqueta	Descripción
getApis	Get	C→H	0x0	Obtiene las API de anfitrión disponibles
getApiVersions	Get	C→H	0x1	Obtiene las versiones disponibles de una API de anfitrión
setApiVersion	Set	C→H	0x2	Fija la versión de la API de Anfitrión a utilizar

9.4.2.2 Mensaje getApis

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

- La respuesta a esta petición es la matriz de bits maxNrApis que indica cuales son las API soportadas por el **Anfitrión ECI**.

Definición de propiedades:

- Disponibilidad de la API de Anfitrión con la etiqueta **a** siendo (**a** < **maxNrApis)** es (**(*result*[a/32]>>(a%32))&0b1 == 0b1)**.

Definición de los parámetros:

maxNrApis: ushort	Número más elevado de las API para el que se devuelve el resultado más uno.

9.4.2.3 Mensaje getApiVersions()

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

- La respuesta a esta petición es una matriz de bits de **maxNrVersions** que indica las versiones de **api** soportadas por el **Anfitrión ECI**.

Definición de propiedades:

- Disponibilidad de la versión de API con etiqueta **api** para la versión **v** siendo (**v** < **maxNrVersions)** es (**(*result*[v/32]>>(v%32))&0b1 == 0b1)**.

Definición de los parámetros:

maxNrVersions: ushort	Número más elevado de versión para el que se devuelve el resultado más uno.

9.4.2.4 Mensaje setApiVersion()

C→H **setApiVersion (**ushort **api**, ushort **version)**

- Este mensaje fija la versión de la API a utilizar entre el **Cliente ECI** y el **Anfitrión ECI**, es decir, el parámetro **version** para el valor **api**. Sólo debe ser llamada una vez (llamadas posteriores no tendrán efecto).

Definición de los parámetros:

api: ushort	Etiqueta de la API para la que se fijará la versión.
version: ushort	Número de versión de **api** a utilizar en sesiones subsiguientes entre cliente y anfitrión.

Información de la Semántica:

- Si **version** no corresponde a una versión de API existente soportada por **api** la versión de la API se fijará en el primer valor superior de versión de API disponible soportada por la propia API o, en otro caso, por el valor más elevado de versión de la API.
- Los **Clientes ECI** comprobarán la disponibilidad de una versión de la API antes de la inicialización a dicha versión de API.

NOTA – Si no se realiza la comprobación la API puede tener un comportamiento imprevisto o pueden producirse errores.

9.4.3 API de acceso a los recursos de la interfaz de Usuario del Anfitrión ECI

9.4.3.1 Introducción

En esta cláusula se define el entorno de aplicación para las aplicaciones **ECI**, que permiten al **Cliente ECI** establecer una interfaz de interacción con el **Usuario**. Las aplicaciones **ECI** se alojan en los **Clientes ECI** y se ejecutan en un **Anfitrión ECI**. Las aplicaciones utilizan un navegador HTML, disponible en dispositivos de TV para una serie de plataformas de fabricantes de dispositivos y radiodifusores.

En la Figura 9.4.3.1-1 se representan las entidades individuales del entorno de aplicación **ECI**. El **Cliente ECI** no controla la aplicación que ha lanzado ni se comunica directamente con ella; utiliza un representante (proxy) facilitado por el **Anfitrión ECI**. El representante implementa la API definida en la cláusula 9.4.3.4 que permite a los **Clientes ECI** lanzar y detener aplicaciones ECI y comunicarse con aplicaciones **ECI** en curso de ejecución, por ejemplo, para el procesamiento de entradas de datos del **Usuario**. El representante gestiona la comunicación de la aplicación **ECI** con el **Cliente ECI** mediante la transcodificación de una petición Get del navegador HTTP en un recurso desde el contenedor de la aplicación o en una petición reqUiClientQuery de la API al **Cliente ECI**, tal como se define en la cláusula 9.4.3.4.8. Esta última puede proporcionar al **Cliente ECI** los datos de entrada del **Usuario** y permitir al **Cliente ECI** responder con un contenido dinámico. El contenedor de la aplicación proporciona los recursos estáticos (de mayor tamaño) para la construcción de los pantallazos de la interfaz de Usuario (UI); el **Cliente ECI** proporciona entradas de datos a medida para la pantalla de la UI y recibe datos del **Usuario**.

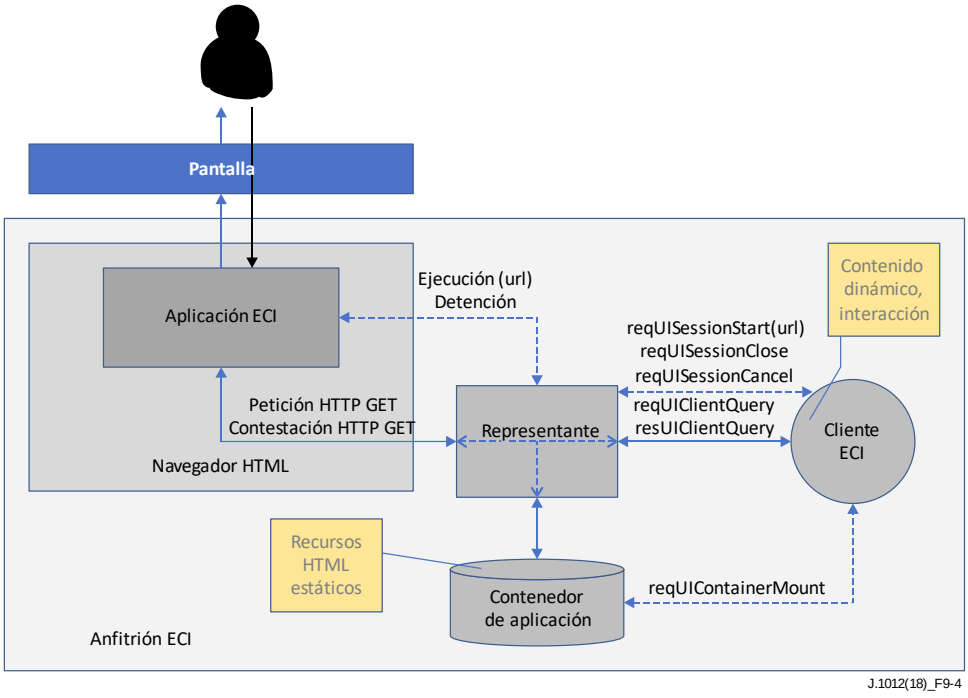


Figura 9.4.3.1-1 – Representación esquemática de la API de la interfaz de Usuario

9.4.3.2 Entorno de la interfaz de Usuario

9.4.3.2.1 Perfil del navegador

El **Anfitrión ECI** proporcionará un navegador HTML que implemente el perfil de internet normalizado para TV definido en [CEI 62766-5-2], que es conforme con las limitaciones y extensiones definidas en la presente Recomendación. El sistema HbbTV [b-HbbTV] también adopta este perfil.

9.4.3.2.2 Limitaciones

El **Anfitrión ECI** denegará las peticiones HTTP a cualquier recurso de una sesión de **Aplicación ECI** no originado en esa sesión de la **Aplicación ECI**.

Los URL utilizados para cargar los recursos de la **Aplicación ECI** en el navegador serán la concatenación de un URL base único para la sesión y un URL relativo para direccionar al **Cliente ECI** o al contenedor de la aplicación. Por ejemplo, si el URL base de la sesión es:

```
http://localhost:3000/session-x/
```

y un recurso en el contenedor de la aplicación es:

```
main/pincode.html
```

el URL del navegador es:

```
http://localhost:3000/session-x/main/pincode.html
```

Cuando atienda peticiones del navegador HTML, el **Anfitrión ECI** debe inferir el tipo de contenido de los recursos de la **Aplicación ECI** a partir de las extensiones de los nombres de los ficheros, debiendo soportar al menos las siguientes:

- text/html – .html y .htm
- text/javascript – .js
- text/css – .css
- image/png – .png
- image/gif – .gif
- image/jpeg – .jpg y .jpeg

9.4.3.2.3 Capacidades del navegador

9.4.3.2.3.1 Modelo de representación

La ventana del navegador ocupará la pantalla completa. La ventana del navegador será de al menos 1 280 x 720 pixeles. Una aplicación **ECI** debe diseñarse de forma que escale adecuadamente para dimensiones mayores.

El plano de gráficos que muestra las aplicaciones **ECI** debe colocarse detrás del plano de gráficos de las aplicaciones del terminal y por delante de cualquier otro plano de gráficos incluidos los de video, los subtítulos y las aplicaciones de difusión.

El plano de las aplicaciones **ECI** cubre por completo cualquier plano de gráficos excepto el del terminal. El fondo de la ventana del navegador debe ser transparente, es decir, si una parte no está ocupada por ningún elemento HTML de la aplicación ECI, deben ser visibles los planos que se encuentran debajo (uno de ellos normalmente contiene el video en difusión). Si se establece que el atributo CSS color de fondo es transparente, la ventana de fondo del navegador será transparente.

Cuando el terminal necesita una superposición temporal sobre la aplicación **ECI**, por ejemplo, para mostrar el menú del sistema o una barra de información del canal en relación con una actuación del **Usuario**, se desenfocará la aplicación **ECI**. Si se desenfoca la aplicación **EC,** se transmitirá un evento desenfocado cuyo objetivo será el objeto Ventana.

Cuando el terminal cierra su interfaz de Usuario (UI) estando en ejecución la aplicación **ECI**, esta vuelve a enfocarse. Si la aplicación **ECI** recibe datos de entrada, se transmitirá un evento enfocado cuyo objetivo será el objeto Ventana. El navegador soportará RGBA32 como formato de color.

9.4.3.2.3.2 Texto y tipografía

El navegador integrará una tipografía proporcional. Las aplicaciones **ECI** pueden seleccionar la tipografía utilizando 'sans-serif' o 'default' como nombres de familias de tipografía genéricas a fin de seleccionar la tipografía integrada. El conjunto de caracteres de la tipografía integrada debe ser la adecuada para la región en la que se despliega el dispositivo. Las aplicaciones **ECI** pueden utilizar la tipografía web CSS3 definida en [CEI 62766-5-2] para tipografías y juegos de caracteres alternativos. El navegador soportará al menos una tipografía en Internet descargable para cada aplicación **ECI**.

El navegador soportará la codificación UTF-8 para todos los recursos de texto de una aplicación **ECI**, es decir, documentos HTML, ficheros de órdenes (scripts) y hojas de estilo.

9.4.3.2.3.3 Formatos de gráficos

El navegador soportará gráficos en los formatos siguientes: GIF [W3C GIF V89a], JPEG [UIT-T T.871] y PNG [W3C PNG].

9.4.3.2.3.4 Entradas de datos del Usuario

El navegador permitirá la entrada de datos de **Usuario** mediante control a distancia utilizando KeyboardEvents DOM3. Cuando una aplicación **ECI** se está ejecutando y la entrada de datos está enfocada, el **Anfitrión ECI** permitirá al Usuario iniciar los siguientes eventos:

- Teclas numéricas: 0-9
- Teclas del cursor: izquierda, derecha, arriba, abajo, entrada y navegación hacia atrás

No se exige soportar los atributos preexistentes keyCode y charCode.

9.4.3.2.3.5 Persistencia

El navegador soportará el almacenamiento de sesión para la API de almacenamiento web (WebStorege) y las cookies de sesión. Un **Cliente ECI** debe utilizar su memoria interna para mantener la información durante varias sesiones del navegador.

9.4.3.2.3.6 Aplicación ECI para acceder a recursos HTML estáticos

El representante (proxy) que recibe las peticiones HTTP originadas por la **Aplicación ECI** establecerá una correspondencia entre el URL relativo (es decir, la extensión desde el URL base de la sesión) y un trayecto relativo en el contenedor de la aplicación organizado por el **Cliente ECI**. La correspondencia entre el URL relativo y el fichero es directa: se hace corresponder el directoryname1/ directoryname2/.. / directorynameN/filename del URL relativo con el nombre del fichero (filename) del directorio directorynameN del directorio contenido en … contenido en el directoryname2 del directorio contenido en el directoryname1 del directorio.

La estructura del directorio del contenedor de la aplicación y los ficheros cumplirán las limitaciones siguientes:

- Todos los nombres de ficheros y directorios constarán de caracteres alfanuméricos y de los caracteres '.' (punto) y '_' (guión bajo) y no tendrán más de 40 caracteres.

En [b-UIT-T J Supl. 7]. se definen recursos o requisitos de calidad de funcionamiento adicionales del contenedor de la aplicación.

9.4.3.2.3.7 Comunicación entre el Cliente ECI y las Aplicaciones ECI

El navegador soporta XmlHttpRequest tal como requiere la cláusula 9.4.3.2.1 de la presente Recomendación. La comunicación entre aplicaciones **ECI** y **Clientes ECI** se encamina a través del representante del **Anfitrión ECI**. La aplicación **ECI** puede realizar una petición Get HTTP utilizando la API XMLHttpRequest tal como se define en esta cláusula. El URL para la petición HTTP se construirá a partir del URL base de la sesión **Aplicación ECI**, tal como se define en la cláusula 9.4.3.2.2 y el URL relativo '/client'. Los parámetros formarán parte de la cadena de consultas en forma de parejas clave-valor. Las claves y los valores sólo constarán de caracteres ASCII. Las claves tendrán una longitud máxima de 31 caracteres y los valores tendrán una longitud máxima de 255 caracteres.

EJEMPLO: http://localhost:3000/ session-20170303-163100-01/client?id=e4f0&p2=v2

Cuando se recibe la petición HTTP el representante en el **Anfitrión ECI** enviará un mensaje reqUiClientQuery al **Cliente ECI** de la aplicación **ECI** tal como se define en la cláusula 9.4.3.4.5 con la cadena de consultas analizada en forma de parejas valor-clave. La contestación del **Cliente ECI** al anfitrión incluirá los parámetros siguientes:

- tipo: una cadena conforme con los tipos de medios tal como definen las normas pertinentes y se documenta en la base de datos IANA de tipos de medios [b-IANA], por ejemplo, aplicaciones/json definidos en [b-IETF RFC 8259];
- código de estado: un entero utilizado en la contestación a la petición Get, es decir, el éxito correspondería al valor 200;
- cuerpo: una cadena de un máximo de 64 kbytes.

Por lo tanto, el **Anfitrión ECI** construirá la contestación a Get HTTP al navegador fijando el valor del encabezamiento Contenido-Tipo al parámetro tipo ('type'), el estado HTTP al valor de error ('error') y el cuerpo de la contestación al valor del parámetro cuerpo ('body').

La comunicación con aplicaciones HTTP no originadas en el **Cliente ECI** queda fuera del alcance de esta versión de la presente Recomendación.

9.4.3.3 Ciclo de vida de la aplicación

9.4.3.3.1 Lanzamiento de una Aplicación ECI

La pantalla de TV es un recurso compartido que utilizan el terminal, el servicio de difusión, el **Operador** y las aplicaciones de terceros. La actual versión de esta Recomendación define un entorno de aplicación para interfaces de **Usuario** básicas necesario para operar un módulo **ECI**, por ejemplo, entrada del PIN, información de suscripción, etc.

La realización de peticiones de **Clientes ECI** apoyándose en **Anfitriones ECI** estará limitada a los casos siguientes:

- El **Anfitrión ECI** va a iniciar una presentación de medios (por ejemplo, después de sintonizar un canal de difusión) que está siendo procesada por el **Cliente ECI**.
- El **Anfitrión ECI** está presentando medios que están siendo procesados por el **Cliente ECI**.
- El **Anfitrión ECI** ha solicitado al **Cliente ECI** que muestre su **Menú de aplicaciones**.
- El **Cliente ECI** indica que desea lanzar una **Aplicación ECI** no relacionada con el tren de flujos, y el **Anfitrión ECI** puede asegurar que el diálogo está relacionado con una petición del **Usuario** o que no está en conflicto con el contenido en pantalla: es decir, no se produce una supresión/pantalla en negro o superposición en pantalla del contenido de un tercero seleccionado para su visionado por el **Usuario**.

A los efectos de lo anterior, una petición de interacción de autenticación parental delegada con el **Usuario**, tal como se define en 9.8.2.11, se considera una petición iniciada por el **Cliente ECI** que emitió la petición de autenticación parental original definida en la cláusula 9.8.2.10.

Un **Conflicto de pantalla** se define como una situación en la que el **Cliente ECI** solicita al **Anfitrión ECI** la ejecución de una **Aplicación ECI** (apertura de una sesión en la UI) sin que se cumplan las condiciones arriba indicadas para iniciar dicha ejecución.

Si el **Anfitrión ECI** tiene la capacidad de ejecutar aplicaciones interactivas, podrá lanzar al menos una **Aplicación ECI** al tiempo que ejecuta dicho contenido interactivo relacionado con los medios presentados en la pantalla. Esa **Aplicación ECI** estará directamente relacionada con los medios presentados en la pantalla. El lanzamiento de la **Aplicación ECI** no da por terminado el contenido interactivo presentado en la pantalla, contenido que podrá retomar la interacción con el **Usuario** cuando se detenga la **Aplicación ECI**.

El **Anfitrión ECI** señalará al **Usuario** la intención de un **Cliente ECI** de iniciar una **Aplicación ECI** no relacionada con el flujo de contenidos o permitirá al **Cliente ECI** iniciar dicha **Aplicación ECI** sin que normalmente se produzca un **Conflicto de pantalla**. Para ello, por ejemplo, las **Aplicaciones ECI** pueden iniciarse inmediatamente después del encendido o en la transición al estado de reposo, o bien el **Usuario** puede tomar alguna medida en contestación a un icono de atención presente en una zona de anuncios o en la pantalla del menú del **Anfitrión ECI** que se muestra habitualmente. Los **Clientes ECI** no deberían asumir la posibilidad de iniciar con frecuencia dichas **Aplicaciones ECI**, debiendo estar estas justificadas por cuestiones importantes para el funcionamiento continuado del **Cliente ECI**.

Cuando el **Cliente ECI** lanza una **Aplicación ECI**, esta se cargará en contextos de navegación no accesibles desde los contextos de navegación de difusión o de cualquier otra aplicación de un tercero.

La ventana del navegador será visible en un plazo de un segundo y deberá haber cargado íntegramente la **Aplicación ECI**.

Versiones futuras de la presente Recomendación podrán proporcionar modelos de un ciclo de vida ampliado y mecanismos de resolución de conflictos, así como permitir la comunicación con aplicaciones HTML lanzadas externamente.

9.4.3.3.2 Finalización de una Aplicación ECI

Para detener una **Aplicación ECI** el **Cliente ECI** transmite un mensaje reqUISessionStop al **Anfitrión ECI**. La petición incluye un uiSessionId enviado por el **Anfitrión ECI** en la contestación resUISessionOpen. La aplicación **ECI** se detendrá. La forma de lograrlo es función de la implementación, por ejemplo, deteniendo o minimizando el navegador. En cualquier caso, la aplicación **ECI** quedará desenfocada y el navegador no enviará KeyboardEvents adicionales a la aplicación **ECI**.

Una aplicación **ECI** también se detendrá si una actuación del **Usuario** (como por ejemplo, pulsar las teclas P+/P-) hace pasar al terminal a un estado que prohíbe el lanzamiento de una aplicación **ECI**. El **Anfitrión ECI** enviará un mensaje reqUiSessionCancel al **Cliente ECI**.

9.4.3.4 Conjunto de las API relacionadas con las comunicaciones de Usuario

9.4.3.4.1 Lista de mensajes de la API de comunicaciones de Usuario

La API de la interfaz de **Usuario** permite al **Cliente ECI** crear un fichero contendor de la aplicación UI descargada que proporcione el grueso de los recursos HTML estáticos necesarios para generar la interfaz de **Usuario**. El representante resuelve automáticamente todas las peticiones HTTP no dirigidas a clientes realizadas desde el navegador al fichero contendor de la aplicación.

El **Anfitrión ECI** puede sugerir al **Cliente ECI** que inicie una aplicación, ya sea en contestación a una solicitud del **Usuario** para el acceso al **Menú de aplicaciones del Cliente ECI** o para indicar al **Cliente ECI** que no existen conflictos que le impidan presentar al **Usuario** una **Aplicación ECI** no relacionada con el **Asa de Medios** mediante el mensaje reqUiSessionCommence. El **Cliente ECI** puede indicar su interés por lanzar ese diálogo no relacionado con el **Asa de Medios** mediante el

mensaje setUiClientAttention. Efectivamente, ello permite establecer comunicaciones de menor prioridad desde el **Cliente ECI** al **Usuario** cuando no existe un **Conflicto de pantalla**.

El **Cliente ECI** abre todas las sesiones de la interfaz de **Usuario** mediante el mensaje reqUiSessionOpen. El URL relativo para la presentación de la primera pantalla de la UI se proporciona como parámetro. Tanto el **Cliente ECI** como el **Anfitrión ECI** pueden terminar la sesión de la interfaz de **Usuario** utilizando los mensajes reqUiSessionClose y reqUiSessionCancel respectivamente.

El mensaje reqUiClientQuery permite a la **Aplicación ECI** del navegador transmitir peticiones con parámetros a través del representante al **Cliente ECI**, que puede responder con datos para la aplicación HTML. Esta comunicación permite a la **Aplicación ECI** presentar al **Cliente ECI** datos específicos y proporcionar al **Cliente ECI** datos de **Usuario** de la misma forma que una aplicación HMTL que se comunica con un servidor HTML dinámico.

En el Cuadro 9.4.3.4.1-1 figuran todas las API definidas en esta cláusula.

Cuadro 9.4.3.4.1-1 – Mensajes de la API de la interfaz de Usuario

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqUiContainerMount	A	C→H	0x0	Configura un contenedor de Aplicación UI con recursos HTML para soportar sesiones UI.
setUiClientAttention	S	C→H	0x1	**El Cliente ECI** indica su intención de iniciar una sesión UI sin asociación con un **Asa de Medios**.
reqUiSessionCommence	A	H→C	0x2	**El Anfitrión ECI** sugiere al **Cliente ECI** que abra una sesión UI.
reqUiSessionOpen	A	C→H	0x3	El **Cliente ECI** solicita la apertura de una sesión de interfaz de **Usuario** con el **Usuario** y presentar el contenido en la pantalla.
reqUiSessionClose	A	C→H	0x4	El **Cliente ECI** termina una sesión de interfaz de **Usuario**.
reqUiSessionCancel	A	H→C	0x5	El **Anfitrión ECI** cancela una sesión de interfaz de **Usuario**.
reqUiClientQuery	A	H→C	0x6	El **Cliente ECI** recibe una petición de la aplicación HTML en el navegador y proporciona una contestación (dinámica).

9.4.3.4.2 Mensaje reqUiContainerMount

C→H reqUiContainerMount(fileName **filename**, PubKey **pk)** →
H→C resUiContainerMount (uint **indexFileLen**, uchar **indexFile)**

- Este mensaje permite al **Cliente ECI** orientar al **Anfitrión ECI** para que identifique un fichero como contenedor de la aplicación del **Cliente ECI** con los recursos HTML para su **Aplicación ECI**. Si tiene éxito, devuelve el contenido del fichero "EciIndex.txt" en el directorio principal del contenedor de la aplicación.

Definición de los parámetros de la Petición:

filename: fileName	Nombre del fichero en el sistema de ficheros del Cliente ECI que será designado contenedor de la aplicación.
pk: PubKey	Clave pública para verificar la firma del contenedor de la aplicación.

Definición de los parámetros de la Contestación:

indexFileLen: uint	Longitud del fichero índice.
indexFile: uchar	Contenido del fichero índice.

Información de la Semántica:

- Los corchetes [and] con un texto prolijo en su interior, tal como se observa más abajo, expresan la demarcación de los campos y estructuras en los contenedores de ficheros ZIP.
- La firma para la verificación del fichero contenedor está en el campo [.ZIP file comment] de la estructura [end of central directory record] (véase la versión 6.3.3, Especificación del formato de fichero Zip, de PKWARE® Inc. al que se hace referencia en [ISO/CEI 21320].

- El [.ZIP Comment Field] se define de forma que *termina* con la cadena siguiente compuesta por todos los caracteres ASCII: 'ECI_SIGNATURE="' seguido del valor de una estructura ECI_Data_Signature como la definida en el Cuadro 5.6-1 codificada como cadena hexadecimal en mayúsculas, seguida de un "" (paréntesis de cierre).

EJEMPLO:

ECI_SIGNATURE="01000000FCB1F60456719035"

La longitud de la cadena de firma de datos codificados para una ECI_Data_Signature de tipo 1 es de 520 caracteres.

- El **Anfitrión ECI** verificará la firma calculada para el fichero contenedor hasta el [.Zip comment field] en la estructura [end of central directory record] y fija el [.Zip comment length field] a 0x0000 utilizando el parámetro de la clave pública pk y el proceso de la cláusula 5.6 definido para el cálculo de firmas.
- El fichero índice se define como el fichero con el nombre "EciIndex.txt" en el directorio principal del fichero contenedor.
- El **Cliente ECI** creará un contenedor de aplicaciones UI válido si es necesario parea las sesiones UI.
- El **Cliente ECI** podrá mostrar un mensaje básico de socorro al **Usuario** en caso de fallo en la carga y creación del contenedor de aplicaciones UI.

Observaciones sobre la aplicación:

- Los clientes pueden cargar ficheros de los contenedores de aplicaciones en su sistema de ficheros desde un servidor en línea utilizando la API HTTP(S) (véase la cláusula 9.4.4.6) o desde un flujo de transporte de difusión utilizando la API del Carrusel de datos.
- El fichero "EciIndex.txt" puede contener información sobre la versión para la UI, verificada mediante la firma de clave pública.

Los códigos de error relativos al mensaje reqUiContainerMount se definen en el Cuadro 9.4.3.4.2-1.

Cuadro 9.4.3.4.2-1 – Códigos de error de reqUiContainerMount

Nombre	Descripción
ErrUiContainerFileNot	Véase el Cuadro 9.4.3.4.9-1.
ErrUiContainerNot	
ErrUiContainerSignature	
ErrUiContainerIndexTxtNot	

9.4.3.4.3 Mensaje setUiClientAttention

C→H setUiClientAttention(uint **clientAttention)**

- Este mensaje indica la intención del **Cliente ECI** de abrir una sesión UI con un **Usuario** que no tiene relación con un **Asa de Medios** (tipo de sesión UI EciUiSessionDiaReq, véase la cláusula 9.4.3.4.4).

Definición de propiedades:

clientAttention: uint	Los valores definidos son: 0x0: no es conveniente la atención del **Usuario**. 0x1: es conveniente la atención del **Usuario**. Todos los demás valores están reservados.

Postcondiciones:

- Si clientAttention=0x0 el **Anfitrión ECI** no generará mensajes reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq).
- Si clientAttention=0x1 el **Anfitrión ECI** generará un mensaje reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq) si no existen mensajes pendientes de este tipo.

9.4.3.4.4 Mensaje reqUiSessionCommence

H→C reqUiSessionCommence (uint **uiSessionType) →**
C→H resUiSessionCommence ()

- Este mensaje permite al **Anfitrión ECI** sugerir al **Cliente ECI** la apertura de una sesión UI de un tipo específico.

Definición de los parámetros de la Petición:

uiSessionType: uint	Nombre del fichero en el sistema de ficheros del **Cliente ECI** que se designará contenedor de la aplicación. Los valores se definen en el Cuadro 9.4.3.4.4-1. Sólo se permiten los valores EciUiSessionAppMenu y EciUiSessionDiaReq.

Cuadro 9.4.3.4.4-1 – Tipos de sesiones UI de la ECI

Nombre	Valor	Descripción
EciUiSessionDiaReq	0x00	El **Cliente ECI** solicitó una sesión UI con el **Usuario** final mediante el mensaje setUiClientAttention (sin asociación a un **Asa de Medios** específica) de forma que a partir de ese momento el **Anfitrión ECI** puede autorizar una reqUISessionOpen desde el **Cliente ECI**.
EciUiSessionAppMenu	0x01	**Menú de aplicación** del **Cliente ECI**. Permite el acceso iniciado por el **Usuario** a todos los valores, información y funciones pertinentes que pueda iniciar el **Usuario**.
EciUiSessionMh	0x02	El **Cliente ECI** ha solicitado una sesión UI asociada a operaciones para un **Asa de Medios**.
EciUiSessionParAuthDel	0x03	El **Cliente ECI** ha solicitado una sesión UI para el diálogo de autenticación parental delegado a los efectos del procesamiento del contenido en un **Asa de Medios**.
RFU	Otros	Reservado para uso futuro.

NOTA – Los valores de Cuadro 9.4.3.4.4-1 se definen siguiendo un orden de prioridad recomendado. Este orden puede ofrecer sugerencias sobre la forma de resolver conflictos de foco de la UI en el diseño del **Anfitrión ECI**.

Información de la Semántica:

- Un **Cliente ECI** podrá presentar un **Menú de Aplicación**. El **Menú de Aplicación** debe permitir, como mínimo, que el **Usuario** inspeccione la versión del **Cliente ECI**, una referencia a la **Operación de Plataforma** y el estado operacional del **Cliente ECI**.

Precondiciones a la Petición:

- No habrá mensajes reqUiSessionCommence pendientes previamente enviados al **Cliente ECI** para una sesión UI.

Postcondiciones a la Contestación:

- El **Cliente ECI** transmitirá un mensaje reqUiSessionOpen con el correspondiente tipo de sesión UI o bien se informa de un error.

Los códigos de error relativos al mensaje reqUiSessionCommence se definen en el Cuadro 9.4.3.4.4-2.

Cuadro 9.4.3.4.4-2 – Códigos de error de reqUiClientSessionCommence

Nombre	Descripción
ErrUiResourceError	Véase el Cuadro 9.4.3.4.9-1.
ErrUiClientError	

9.4.3.4.5 Mensaje reqUiSessionOpen

C→H reqUiSessionOpen(uint **uiSessionType**, ushort **mH**, uint **relUrlLen**, char **relUrl[]) →**
H→C resUiSessionOpen(ushort **uiSessionId)**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** una nueva sesión UI.

Definición de los parámetros de la Petición:

uiSessionType: uint	Tipo de sesión UI como se define en el Cuadro 9.4.3.4.4-1. El parámetro mH será pertinente si el valor es EciUiSessionMh o EciUiSessionParAuthDel, pero no en cualquier otro caso.
mH: ushort	Asa de Medios de la sesión de procesamiento del contenido con la que el MMI está asociado.
relUrlLen: uint	Longitud en bytes de relUrl
relUrl: char[]	URL relativo, terminado por un carácter nulo. Añadido al URL base de la sesión conformará el URL para que el navegador inicie la sesión UI. Véase la cláusula 9.4.3.2.2.

Definición de los parámetros de la Contestación:

uiSessionId: ushort	ID de la nueva sesión UI.

Información de la Semántica:

- Un **Cliente ECI** podrá mantener varias sesiones UI simultáneamente. No obstante, solo se es necesario soportar simultáneamente una sesión de tipo de sesión UI EciUiSessionAppMenu o EciUiSessionAppMenu, y como máximo una sesión de tipo de sesión UI EciUiSessionMh por cada **Asa de Medios** abierta.
- Un **Cliente ECI** podrá abrir simultáneamente varias sesiones UI de tipo EciUiSessionMh.
- El **Cliente ECI** podrá abrir simultáneas varias sesiones UI de tipo de sesión UI EciUiSessionParAuthDel si el **Cliente ECI** soporta la API de delegación de la autenticación parental. Dichas sesiones UI podrán ejecutarse en paralelo con otras sesiones UI del **Cliente ECI**.
- Un **Anfitrión ECI** puede mantener una o más sesiones UI simultáneas según sean los modos de aplicación de sus **CPE**.

Precondiciones a la Petición:

1) Si el valor de uiSessionType es EciUiSessionAppMenu o EciUiSessionDiaReq el mensaje estará precedido por un mensaje reqUiClientCommence con el mismo parámetro uiSessionType.
2) Si el valor de uiSessionType es EciUiSessionParAuthDel el mensaje estará precedido por un mensaje reqParAuthDel para el distintivo de medios mH desde el **Anfitrión ECI** al **Cliente ECI**.

3) Si el valor de uiSessionType es EciUiSessionMh, Mh será una sesión de distintivo de medios abierta.

Precondiciones a la Contestación:

1) Si el valor de uiSessionType es EciUiSessionAppMenu, EciUiSessionDiaReq o EciUiSessionParAuthDel el **Anfitrión ECI** solo aceptará la petición de sesión UI si ha sido solicitada previamente, no han desaparecido los motivos que justificaron la petición y se encuentra en un estado que no genera un **Conflicto de pantalla**.
2) Si el valor de uiSessionType es EciUiSessionMh el **Anfitrión ECI** concederá la petición de sesión UI si puede establecer una interacción de interés con el **Usuario** sin generar un conflicto de prioridad de pantalla.
3) Los **Anfitriones ECI** no rechazarán una segunda sesión de un **Cliente ECI** si el uiSessionType de la misma es EciUiSessionParAuthDel. El **Anfitrión ECI** puede cancelar la primera sesión.

Notas de aplicación:

1) El **Anfitrión ECI** rechazará una sesión del **Asa de Medios** si esta se utiliza para registro sin que pueda iniciarse un diálogo con el **Usuario** porque ello daría lugar a un **Conflicto de pantalla** o por no haber una pantalla activa.
2) Se recomienda que las aplicaciones del **Anfitrión ECI** incluyan sesiones UI de autenticación parental cuando, por ejemplo, se programen grabaciones futuras que puedan requerir la autenticación del control parental mediante el mensaje reqParAuthCid de la API de autenticación parental (véase 9.8.2.10).
3) Los **Anfitriones ECI** pueden cancelar una sesión UI con un **Cliente ECI** para permitir una nueva sesión con uiSessionType igual a EciUiSessionParAuthDel o EciUiSessionMh.

Los códigos de error relativos al mensaje reqUiSessionOpen se definen en el Cuadro 9.4.3.4.5-1.

Cuadro 9.4.3.4.5-1 – Códigos de error de reqUiClientSessionStart

Nombre	Descripción
ErrUiScreenConflict	Véase el Cuadro 9.4.3.4.9-1.
ErrUiNoScreen	

9.4.3.4.6 Mensaje reqUiSessionClose

C→H reqUiSessionClose(ushort **uiSessionId) →**
H→C resUiSessionClose(ushort **uiSessionId)**

- Este mensaje permite al **Cliente ECI** cerrar una sesión UI existente.

Definición de los parámetros de la Petición:

uiSessionId: ushort	ID de la sesión UI a cerrar.

Definición de los parámetros de la Contestación:

uiSessionId: ushort	ID de la sesión UI que ha sido cerrada.

Precondiciones a la Petición:

1) Se abrirá una sesión UI con uiSessionId.
2) No se enviarán al **Anfitrión ECI** mensajes adicionales que hagan referencia a uiSessionId.

Precondiciones a la Contestación:

1) No se enviarán al **Cliente ECI** mensajes adicionales que hagan referencia a uiSessionId.

9.4.3.4.7 Mensaje reqUiSessionCancel

H→C reqUiSessionCancel (ushort **uiSessionId**, uint **reason) →**
C→H resUiSessionCancel (ushort **uiSessionId)**

- Este mensaje permite al **Anfitrión ECI** cerrar una sesión UI existente con un **Cliente ECI**. Este mensaje está destinado a ser utilizado por el **Anfitrión ECI** en casos en que hayan dejado de cumplirse las condiciones para un **Aplicación ECI**, por ejemplo, si un **Usuario** cambia a otro canal de un **Cliente ECI** diferente, dando lugar a un **Conflicto de Pantalla**.

Definición de los parámetros de la Petición:

uiSessionId: ushort	ID de la sesión UI a cancelar.
reason: uint	Motivo de la cancelación de la sesión. Los valores se definen en el Cuadro 9.4.3.4.9-1.

Definición de los parámetros de la Contestación:

uiSessionId: ushort	ID de la sesión UI que ha sido cancelada.

Precondiciones a la Petición:

1) La sesión se abrirá con uiSessionId.
2) No se enviarán mensajes adicionales que hagan referencia a uiSessionId.

Precondiciones a la Contestación:

1) No se enviarán mensajes adicionales que hagan referencia uiSessionId.

9.4.3.4.8 Mensaje 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[])**

- Este mensaje transporta una petición HTTP de la **Aplicación ECI** que se está ejecutando en el navegador del **Anfitrión ECI** tal como se describe en la cláusula 9.4.3.2.3.7 y permite que el **Cliente ECI** devuelva una contestación HTTP a la **Aplicación ECI**.

Definición de los parámetros de la Petición:

uiSessionId: ushort	Id de la sesión UI desde la que se hace la petición.
queryLen: uint	Longitud en bytes del parámetro de consulta.
query[]: KeyValPair	Contiene pares de valores clave de los parámetros de consulta de la petición HTTP realizada por el navegador.

Definición de tipos de KeyValPair

```
#define MaxKeyLen 32
#define MaxValLen 256

typedef struct KeyValPair {
   char key[MaxKeyLen];   /* Clave de la pareja de valores de clave, terminado
con nulo */
   char val[MaxValLen];   /* Valor de la pareja de valores de clave, terminado
con nulo */
} KeyValPair;
```

Definición de los parámetros de la Contestación:

uiSessionId: ushort	Id de la sesión UI.
statusCode: uint	Código de estado HTTP definido en [IETF RFC 7231].
typeLen: uint	Longitud en bytes del parámetro tipo.
type[]: char	Tipo de contestación en forma de cadena de caracteres ASCII terminada con un valor nulo.
bodyLen: uint	Longitud en bytes del parámetro cuerpo (body).
body[]: uchar	Mensaje-contestación HTTP.

Precondiciones a la Petición:

1) **uiSessionId** está abierto.

Información de la Semántica:

- Si el formato de la cadena de consulta de la **Aplicación ECI** es deficiente, el **Anfitrión ECI** puede devolver el código de estado HTTP 400 y no iniciar una petición con el **Cliente ECI**.
- En la cláusula 9.4.3.2.3.7 se define la relación de los parámetros del mensaje con la petición y la contestación HTTP del navegador.

9.4.3.4.9 Códigos de error de la API de comunicaciones de Usuario

Los códigos de error relacionados con las comunicaciones en la interfaz de **Usuario** figuran en el Cuadro 9.4.3.4.9-1.

Cuadro 9.4.3.4.9-1 – Códigos de error de la API para la comunicaciones de Usuario

Nombre	Valor	Descripción
ErrUiContainerFileNot	-256	No se ha encontrado un fichero contendor de la aplicación UI.
ErrUiContainerNot	-257	El fichero no es un fichero contenedor de aplicación UI válido.
ErrUiContainerSignature	-258	Fallo en la verificación de la firma del fichero contenedor de la aplicación.
ErrUiContainerIndexTxtNot	-259	No existe un fichero "EciIndex.txt" en el directorio de mayor nivel del contenedor de la aplicación.
ErrUiResourceError	-260	El **Cliente ECI** no puede configurar el recurso contenedor de aplicación UI.
ErrUiClientError	-261	El **Cliente ECI** no está en un estado operacional en el que pueda presentar una UI.
ErrUiDiaNoMore	-262	La petición de diálogo del **Cliente ECI** ya no es válida.
ErrUiScreenConflict	-263	El **Anfitrión ECI** tiene un **Conflicto de Pantalla** y no puede incluir o mantener una sesión.
ErrUiNoScreen	-264	El **Anfitrión ECI** no tiene o ha dejado de tener acceso a una pantalla para la presentación de la sesión UI.
RFU	Otros	Reservado para uso futuro.

9.4.4 API de acceso a los recursos de la pila IP del Anfitrión ECI

9.4.4.1 Introducción

En los **CPE** equipados con una pila IP, el **Anfitrión ECI** proporciona un servicio de acceso a Internet en nombre de **Clientes ECI**. Los **Clientes ECI** pueden transmitir mensajes utilizando conexiones UDP/IP y TCP/IP establecidas con sus pares tanto en modo **Cliente ECI** como servidor utilizando los **Anfitriones ECI**. Los nombres de los **Anfitriones ECI** pueden resolverse en direcciones IP mediante los servicios DNS disponibles en el **Anfitrión ECI**.

La seguridad de los servicios proporcionados está limitada por las capacidades de seguridad genéricas del software del propio **CPE**. Es decir, si el software del **CPE** ajeno al **Anfitrión ECI** se ve comprometido, podría manipularse cualquier tráfico IP.

La API del **Cliente ECI** para conectividad IP se basa en el paradigma del conector BSD tal como se utiliza en numerosos sistemas operativos actuales.

La definición de la API se divide en cuatro partes:

1) Conectores IP **ECI** básicos y funcionalidad DNS (cláusula 9.4.4.3).
2) Comunicaciones UDP/IP utilizando un conector IP **ECI** (cláusula 9.4.4.4).
3) Comunicaciones TCP/IP utilizando un conector IP **ECI** (cláusula 9.4.4.5).
4) Comunicaciones HTTP(S) utilizando los servicios HTTP del **Anfitrión ECI** (cláusula 9.4.4.6).

9.4.4.2 Especificaciones básicas

Un **Anfitrión ECI** con capacidad de conexión IP implementará el protocolo [IETF RFC 791] incluido IPv6 [IETF RFC 8200] y las actualizaciones aplicables. Proporcionará la forma de resolver el nombre del **Anfitrión ECI** en direcciones IP que utilizan el DNS con arreglo a [IETF RFC 1034], [IETF RFC 1035] y las actualizaciones aplicables de los mismos.

Para la provisión de un protocolo sencillo de mensajes cortos sin garantías el Anfitrión ECI deberá soportar UDP sobre IP de conformidad con [IETF RFC 768] incluidas las actualizaciones aplicables. Para la provisión del intercambio de mensajes orientados a la conexión con garantías de entrega el Anfitrión ECI soportará TCP sobre IP de conformidad con [IETF RFC 793] incluidas las actualizaciones aplicables.

No es necesario que el **Anfitrión ECI** soporte la multidifusión UDP en modo transmisión o recepción.

9.4.4.3 Conectores IP de la ECI

9.4.4.3.1 Generalidades

Los **Cliente ECI** pueden abrir un conector IP de la ECI para transmisión o recepción utilizando TCP e IP.

NOTA – El término "conector" ("socket") evoca una similitud con los conectores BSD originales de muchos sistemas operativos. Conceptualmente, los conectores IP de la **ECI** son similares a los conectores BSD pero tienen características específicas que los diferencian. En particular, el comportamiento es completamente asíncrono.

Los conectores IP de la **ECI** son puntos extremos de las comunicaciones IP. Los **Clientes ECI** pueden abrir un conector identificando el número del puerto local y aceptando las **Peticiones** de conexión entrantes (operación similar a la de un servidor TCP/IP). Los conectores pueden estar cerrados, en cuyo caso cualquier conexión o comportamiento de servidor asociado también estará cerrado. La dirección IP del nombre de un anfitrión par puede resolverse mediante los servicios DNS del **Anfitrión ECI**.

Los mensajes disponibles figuran en el Cuadro 9.4.4.3.1-1.

Cuadro 9.4.4.3.1-1 – Mensajes de conectores IP

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqIpSocket	A	C→H	0x0	Abre un conector IP de la **ECI**.
reqIpClose	A	C→H	0x1	Cierra un conector IP de la **ECI**.
reqIpAddrinfo	A	C→H	0x2	Obtiene la dirección del **Anfitrión ECI** (distante).

Las definiciones del tipo de estructura de estas API se definen en la cláusula 9.3.

Definición de tipos de la API de conector IP:

```
typedef struct Addrinfo {
   ushort addressType;       /* dirección IPv4 o IPv6 */
   uchar  ipAddress[16];     /* dirección IP */
   ushort port;              /* número de puerto – si es
                    pertinente */
} Addrinfo;
```

Definición de campos:

addressTyp: ushort.	Véase el Cuadro 9.4.4.3.4-1, solo se permiten los valores ProtPrefIPv4 o ProtPrefIPv6. Este campo define que la longitud de hosAddress (dirección de anfitrión) es de 4 ó 16 bytes (véase la nota).
ipAddress: uchar[16]	4 ó 16 bytes que representa en bytes (en el orden de red) una dirección IPv4 o IPv6 respectivamente. Las direcciones IPv4 utilizarán los primeros 4 bytes de este parámetro.
port: ushort	Número de puerto del conector al que conectarse (el campo puede no estar utilizado).
NOTA – **ProtPrefIPv4** o **ProtPrefIPv6** se definen en el Cuadro 9.4.4.3.4-1.	

9.4.4.3.2 Mensaje reqIpSocket

C→H reqIpSocket(uchar **source**, ushort **sourcePort**, ushort **protocol) →**
H→C resIpSocket(uchar **socketId)**

- Este **mensaje** abre un conector para comunicaciones TCP o UDP en una dirección y puerto IP local.

Definición de los parámetros de la Petición:

source: uchar	Véase el Cuadro 9.4.4.3.2-1: especifica la dirección IP del Anfitrión ECI que debe utilizarse para el conector local (una preferencia en caso de asignación de varias direcciones IP). Si la dirección IP específica no es identificable, el **Anfitrión ECI** seleccionará una alternativa adecuada.
sourcePort: ushort	Dirección del puerto del punto extremo de la conexión IP local. Un valor 0x0000 significa que el **Anfitrión ECI** asignará una dirección de puerto libre para el conector. No están permitidos otros valores inferiores a 1024.
Protocol: ushort	Véase el Cuadro 9.4.4.3.2-2: especifica el protocolo utilizado para el conector. Específicamente se seleccionará IPv4 o IPv6.

Cuadro 9.4.4.3.2-1 – Parámetros del origen IP

Nombre	Valor	Descripción
IpSourceAny	0x00	Dirección IP por defecto del **Anfitrión ECI**.
IpSourceWan	0x01	Dirección IP del **Anfitrión ECI** utilizada para las comunicaciones en la WAN (internet).
IpSourcePriv	0x02	Dirección IP del **Anfitrión ECI** utilizada para tráfico IP privado en un canal con un protocolo IP patentado.
IpSourceLan	0x03	Dirección IP del **Anfitrión ECI** utilizada para las comunicaciones en la red local.
RFU	Otros	Reservado para uso futuro.

Cuadro 9.4.4.3.2-2 – Parámetros del protocolo IP

Nombre	Valor	Descripción
SockProtUdpIPv4	0x0001	UDP/IP que utiliza IPv4.
SockProtUdpIPv6	0x0002	UDP/IP que utiliza IPv6.
SockProtUdpIPany	0x0003	UDP/IP que utiliza IPv4 o v6.
SockProtTcpClientIpv4	0x0005	TCP/IP que utiliza IPv4, modo cliente (solo para iniciar la conexión).
SockProtTcpClientIpv6	0x0006	TCP/IP que utiliza IPv6, modo cliente (solo para iniciar conexiones)
SockProtTcpClientIpany	0x0007	TCP/IP que utiliza IPv4 o v6, modo cliente (solo para iniciar conexiones)
SockProtTcpServerIpv4	0x0009	TCP/IP que utiliza IPv4, modo servidor (para aceptar conexiones entrantes).
SockProtTcpServerIpv6	0x000A	TCP/IP que utiliza IPv6, modo servidor (para aceptar conexiones entrantes).
SockProtTcpServerIpany	0x000B	TCP/IP que utiliza IPv4 o IPv6, modo servidor (para aceptar conexiones entrantes).
RFU	otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

SocketId: uchar	ID de conector del conector abierto.

Descripción Semántica:

- Inmediatamente después de la inicialización es posible retener la **Contestación** hasta que se haya completado satisfactoriamente la inicialización de la dirección IP del **Anfitrión ECI**. Las cifras de calidad de funcionamiento se proponen en [b-UIT-T J Supl. 7].

Precondiciones a la Petición:

1) No se superará el número máximo de conectores que se permite que solicite el **Cliente ECI**.
2) El origen, el sourcePort (puerto de origen) y el protocolo ha de ser una configuración de parámetros válida.

Postcondiciones a la Contestación:

1) El conector está abierto o bien se devuelve un error en la **Contestación**.

Los códigos de error relativos a la apertura de los conectores figuran en el Cuadro 9.4.4.3.2-3.

Cuadro 9.4.4.3.2-3 – Códigos de error de resIpSocket

Nombre	Descripción
ErrIpSourceProt	Véase el Cuadro 9.4.4.7-1.
ErrIpNoSockets	
ErrIpProtNotAvail	
ErrIpPortNotAvail	

9.4.4.3.3 Mensaje reqIpClose

C→H reqIpClose(uchar **socketId) →·**
H→C resIpClose(uchar **socketId)**

- Cierra el conector IP y cualquier conexión asociada; pueden perderse todas las comunicaciones pendientes hacia/desde el conector.

Definición de los parámetros de la Petición:

socketId: uchar	ID del conector a cerrar.

Definición de los parámetros de la Contestación:

socketId: uchar	ID del conector cerrado.

Descripción Semántica:

- Esta **Petición** cierra el conector y cualquier conexión IP asociada al mismo. Deja al **Anfitrión ECI** el envío de los mensajes de desconexión adecuados al par que proceda. La compleción exitosa de lo anterior no es condición necesaria para transmitir la **Contestación**. También se cerrará cualquier conector que no tenga una conexión asociada.

Precondiciones:

1) El conector existe y su estado es abierto.

Postcondiciones:

1) El conector está cerrado y no puede utilizarse para ninguna comunicación (salvo que sea reasignado con reqIpSocket).

Los códigos de error relativos al cierre del conector figuran en el Cuadro 9.4.4.3.3-1.

Cuadro 9.4.4.3.3-1 – Códigos de error de resIpClose

Nombre	Descripción
ErrIpSocketNotOpen	Véase el Cuadro 9.4.4.7-1.

9.4.4.3.4 Mensaje reqIpAddrInfo

C→H reqIpAddrinfo(uint **hostnameLenth**, char **hostname[]**, uchar **protPref) →**
H→C resIpAddrinfo(Addrinfo **ipaddress)**

- Este mensaje proporciona la información sobre la dirección IP para el direccionamiento del **Anfitrión ECI** utilizando el protocolo preferido (protPref), y devuelve la dirección del **Anfitrión ECI**. El protocolo utilizará los servicios DNS del **Anfitrión ECI** cuando sea necesario para resolver la **Petición**.

Definición de los parámetros de la Petición:

hostNameLength: uint	Longitud (en bytes) del campo nombre.
hostname: char[]	Nombre del anfitrión IP que debe resolverse; ya sea en notación IPv4 con puntos [IETF RFC 952], notación IPv6 con dos puntos [IETF RFC 8200] o el nombre del anfitrión real [IETF RFC 1123].
protPref: uchar	Indica la preferencia del protocolo IP tal como se define en el Cuadro 9.4.4.3.4-1.

Cuadro 9.4.4.3.4-1 – Parámetros de preferencia de protocolo IP

Nombre	Valor	Descripción
ProtPrefIpv4	0x1	Se devolverá una dirección IPv4.
ProtPrefIPv6	0x2	Se devolverá una dirección IPv6.
ProtPrefAny	0x3	Se devolverá una dirección IPv4 o IPv6.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

Ipaddress: Addrinfo	Dirección IP del **Anfitrión ECI**. El campo puerto no está definido.

Descripción Semántica:

- Esta **Petición** utiliza los servicios DNS del **Anfitrión ECI** para traducir el nombre del anfitrión facilitado a una representación binaria de la dirección del anfitrión. Pueden producirse demoras por la indisponibilidad temporal del acceso al servicio DNS (por ejemplo, durante el arranque del **CPE**); el **Anfitrión ECI** garantizará que se respeta la temporización (por ejemplo, el **Cliente ECI** siempre recibe la **Contestación**).

Postcondiciones a la Contestación:

1) Dirección del anfitrión resuelta o bien se produce un error.

Los códigos de error relativos al cierre del conector figuran en el Cuadro 9.4.4.3.4-2.

Cuadro 9.4.4.3.4-2 – Códigos de error de resIpAddrInfo

Nombre	Descripción
ErrIpHostUnknown	Véase el Cuadro 9.4.4.7-1.
ErrIpHost	
ErrDnsOffline	

9.4.4.4 UDP/IP de la ECI

9.4.4.4.1 Generalidades

Los **Clientes ECI** enviarán y recibirán datagramas UDP utilizando un conector UDP/IP de la **ECI** abierto. Los mensajes conexos se definen en el Cuadro 9.4.4.4.1-1.

Cuadro 9.4.4.4.1-1 – Mensajes UDP/IP en el conector

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqIpUdpSendMsg	A	C→H	0x3	Envía el mensaje al puerto UDP par.
reqIpUdpRecvMsg	A	C→H	0x4	Recibe el mensaje del puerto UDP par.

9.4.4.4.2 Mensaje reqIpUdpSendMsg

C→H reqIpUdpSendMsg(uchar **socketId,** Addrinfo **peer**, uint **datagramLength**, byte **datagram[]) →**
H→C resIpUpdSendMsg(uchar **socketId)**

- Este mensaje envía un datagrama UDP a un par (dirección IP, puerto IP).

Definición de los parámetros de la Petición:

socketId: uchar	Longitud (en bytes) del campo nombre.
peer: Addrinfo	Par (dirección IP, número de puerto IP) destino del datagrama.
datagramLength: uint	Longitud (en bytes) del datagrama.
datagram: byte[]	Contenido del datagrama (bytes en el orden de la red).

Definición de los parámetros de la Contestación:

socketId: uchar	Conector desde el que se envió la correspondiente **Petición**.

Descripción Semántica:

- El datagrama se envía al par utilizando el protocolo UDP y la dirección y puerto del anfitrión IP del conector.

Precondiciones a la Petición:

1) El conector había sido abierto para UDP utilizando la misma estructura de dirección que el par.

Postcondiciones:

1) Se ha enviado el datagrama (aunque puede perderse).

Los códigos de error relativos al envío de datagramas UDP figuran en el Cuadro 9.4.4.4.2-1.

Cuadro 9.4.4.4.2-1 – Códigos de error de resIpUdpSendMsg

Nombre	Descripción
ErrIpUdpProtMismatch	Véase el Cuadro 9.4.4.7-1.
ErrIpUdpSocketNot	
ErrIpUdpTooLong	
ErrIpUdpIpOffline	

9.4.4.4.3 Mensaje reqIpUdpRecvMsg

C→H reqIpUdpRecvMsg(uchar **socketId) →**
H→C resIpUdpRecvMsg(uchar **socketId**, Addrinfo **peer**, uint **datagramLength**,
byte **datagram[])**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** que reciba un datagrama UDP desde un par (es decir, nombre de anfitrión, puerto) enviado al conector con **SocketId**.

Definición de los parámetros de la Petición:

socketId: uchar	Conector (con número de puerto y dirección de anfitrión) el que está previsto recibir el datagrama.

Definición de los parámetros de la Contestación:

socketId: uchar	Longitud en bytes del campo nombre (name).
peer: Addrinfo	Dirección IP + número de puerto origen del datagrama (par).
datagramLength: uint	Longitud (en bytes) del datagrama.
datagram: byte[]	Contenido del datagrama (bytes en el orden de la red).

Descripción Semántica:

- Un datagrama puede recibirse en el conector, en cuyo caso se devuelve una **Contestación**.

NOTA 1 – El cierre del conector dará por terminada cualquier **Petición reqIpUdpRecvMsg** pendiente.

NOTA 2 – Está permitido el envío de varios **reqIpUdpRecvMsg** antes de recibir las correspondientes **Contestaciones** en el mismo conector, aunque el **Anfitrión ECI** no tiene la obligación de permitir el encolamiento de más de cinco de dichas **Peticiones**.

Precondiciones a la Petición:

- El conector se ha abierto para UDP.

Postcondiciones a la Contestación:

- Se ha enviado el datagrama (aunque puede perderse).

Los códigos de error relativos a la recepción de datagramas UDP figuran en el Cuadro 9.4.4.4.3-1.

Cuadro 9.4.4.4.3-1 – Códigos de error de resIpUdpRecvMsg

Nombre	Descripción
ErrIpUdpSocketNot	Véase el Cuadro 9.4.4.7-1.

9.4.4.5 TCP/IP de la ECI

9.4.4.5.1 Generalidades

Los **Clientes ECI** pueden transmitir y recibir mensajes en una conexión TCP/IP abierta relativos a la creación de un conector, dando lugar a una secuencia de flujo de bytes bidireccional libre de errores desde el **Cliente ECI** local a un servicio del par distante o viceversa. Esto permite al **Cliente ECI** actuar como un servidor para las **Peticiones** de canales de terceros (normalmente para aplicaciones LAN). Los mensajes figuran en el Cuadro 9.4.4.5.1-1.

Cuadro 9.4.4.5.1-1 – Mensajes del conector TCP/IP

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqIpTcpConnect	A	C→H	0x5	El cliente TCP se conecta al par servidor TCP.
reqIpTcpSend	A	C→H	0x6	Envío de datos al par conectado.
reqIpTcpRecv	A	C→H	0x7	Recepción de datos del par conectado.
reqIpTcpAccept	A	C→H	0x8	El par servidor TCP acepta la conexión del par cliente TPC.

9.4.4.5.2 Mensaje reqIpTcpConnect

C→H reqIpTcpConnect(uchar **socketId**, Addrinfo **peer)** →
H→C resIpTcpConnect(uchar **socketId)**

- Este mensaje solicita al **Anfitrión ECI** la apertura de una conexión desde un conector TCP abierto al par utilizando el protocolo del conector.

Definición de los parámetros de la Petición:

socketId: uchar	Conector (implica el número de puerto y la dirección del anfitrión) desde el que debe establecerse la conexión TCP.
peer: Addrinfo	Dirección IP y puerto IP del par con el que debe abrirse la conexión.

Definición de los parámetros de la Contestación:

socketId: uchar	ID de conector del conector activado de la **Petición**.

Descripción Semántica:

- El anfitrión local intentará abrir una conexión TCP desde el conector local al par (dirección IP, puerto IP).

Precondiciones:

- El conector ha sido abierto para TCP utilizando el mismo tipo de dirección IP (IPv4 o IPv6) que peerAddressType.

Postcondiciones:

- Se ha establecido la conexión TCP o bien se devuelve una situación de error.

Los códigos de error relativos a la conexión a través de TCP e IP figuran en el Cuadro 9.4.4.5.2-1.

Cuadro 9.4.4.5.2-1 – Códigos de error de resIpTcpConnect

Nombre	Descripción
ErrIpTcpProtMismatch	Véase el Cuadro 9.4.4.7-1.
ErrIpTcpSockNot	
ErrIpTcpIpOffline	
ErrIpTcpConnRefused	
ErrIpTcpConnTimeout	

9.4.4.5.3 Mensaje reqIpTCPSend

C→H reqIpTcpSend(uchar **socketId**, bool **more**, uint **dataLen**, byte **data[])** →
H→C resIpTcpSend(uchar **socketId**, uint **actLen)**

- Este mensaje permite transmitir datos utilizando TCP sobre un conector TCP conectado.

Definición de los parámetros de la Petición:

socketId: uchar	Conector (implica número de puerto y dirección de anfitrión) utilizado para transmitir los datos al par.
more: booleano	Indica si los datos y los datos precedentes deben enviarse al par inmediatamente (**more**=**Falso**) o si se enviarán más datos en posteriores **Peticiones** reqipTcpSend (**more**=**Verdadero**).
dataLen: uint	Cantidad de datos a transmitir.
data: byte[]	Datos a transmitir.

Definición de los parámetros de la Contestación:

socketId: uchar	ID de conector del conector sobre el que se realiza la transmisión.
actLen: uint	Número real de bytes transmitidos satisfactoriamente.

Descripción Semántica:

- El anfitrión local transmitirá los **datos** al par conectado a través de un conector TCP/IP conectado con **socketID**.

Precondiciones a la Petición:

1) El conector está en modo TCP/IP conectado.

Postcondiciones a la Contestación:

1) Si actLen no es igual a dataLen se producirá un error.

Los códigos de error relativos a la transmisión de paquetes TCP figuran el Cuadro 9.4.4.5.3-1.

Cuadro 9.4.4.5.3-1 – Códigos de error de resIpTcpSend

Nombre	Descripción
ErrIpTcpSockNot	Véase el Cuadro 9.4.4.7-1.
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.4 Mensaje reqIpTCPRecv

C→H reqIpTcpRecv(uchar **socketId**, uint **maxDataLen) →**
H→C resIpTcpRecv(uchar **socketId**, uint **dataLength**, byte **data[])**

- Este mensaje permite recibir datos utilizando TCP sobre un conector TCP conectado.

Definición de los parámetros de la Petición:

socketId: uchar	Conector (implica número de puerto y dirección de anfitrión) utilizado para la recepción de los datos dirigidos al par.
maxDataLen: uint	Cantidad máxima de datos a recibir.

Definición de los parámetros de la Contestación:

socketId: uchar	ID de conector del conector en el que se generó el mensaje recibido.
dataLength: uint	Número de bytes de datos recibidos desde el par.
data: byte[]	Datos tal como han sido recibidos desde el par.

Descripción Semántica:

- El anfitrión local recibe **datos** desde el par sobre un conector TCP/IP conectado cuyo identificador es **socketID**.

Precondiciones a la Petición:

1) El conector es un conector TCP.

Postcondiciones a la Contestación:

1) Todos los datos disponibles hasta la longitud se devuelven hasta el campo **maxDataLen** en la **Petición**. Si no hay datos disponibles, la **Contestación** se detiene hasta que se cierra la conexión, la conexión TCP se considere temporalmente indisponible o se haya perdido la conexión local con la red IP.

Los códigos de error relativos a la recepción de paquetes TCP figuran en el Cuadro 9.4.4.5.4-1.

Cuadro 9.4.4.5.4-1 – Códigos de error de resIpTcpRecv

Nombre	Descripción
ErrIpTcpSockNot	Véase el Cuadro 9.4.4.7-1.
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.5 Mensaje reqIpTCPAccept

C→H reqIpTcpAccept(uchar **socketId) →**
H→C resIpTcpAccept(uchar **socketId**, uchar **newSocketId**, Addrinfo **peer)**

- Este mensaje acepta una **Petición** de conexión entrante sobre un conector del servidor TCP. Se atenderán las **Peticiones** de conexión pendientes hasta un máximo definido por la implementación del **Anfitrión ECI**. En [b-UIT-T J Supl. 7] se definen los requisitos de calidad de funcionamiento del servidor TCP.

Definición de los parámetros de la Petición:

socketId: uchar	Conector (implica número de puerto y dirección del anfitrión) utilizado para la recepción de **Peticiones** de conexión.

Definición de los campos del mensaje:

socketId: uchar	ID de conector del conector en el que se realizó la petición.
newSocketId: uchar	ID de conector de la conexión recién abierta con el par que transmitió una **Petición** de conexión. La dirección del anfitrión y el puerto se heredan del conector con **socketId**.
peer: Addrinfo	Dirección IP + Puerto IP del par en la conexión.

Descripción Semántica:

- El **Anfitrión ECI** local espera las **Peticiones** de conexión TCP entrantes sobre la dirección IP/puerto especificada en la creación del conector y abre un nuevo conector conectado que atienda cada **Petición** de conexión entrante (o pendiente). No habrá **Contestación** si no existe un **Petición** entrante o si el conector servidor está cerrado.

Precondiciones a la Petición:

1) El conector es un conector servidor TCP.

Postcondiciones a la Contestación:

1) Ante cualquier **Petición** de conexión al conector servidor, se devuelve un nuevo conector con una conexión TCP/IP abierta, o bien se produce un error.

Los códigos de error relativos a la aceptación de conexiones TCP figuran en el Cuadro 9.4.4.5.5-1.

Cuadro 9.4.4.5.5-1 – Códigos de error de resIpTcpAccept

Nombre	Descripción
ErrIpTcpListSockNot	Véase el Cuadro 9.4.4.7-1.
ErrIpTcpNoMoreSockets	

9.4.4.6 API para servicios Get HTTP(S)

9.4.4.6.1 Generalidades

El **Anfitrión ECI** proporcionará peticiones básicas GET HTTP(S) para obtener recursos de un servidor IP HTTP. Ello permite al **Cliente ECI** obtener recursos en Internet (ficheros) de servidores en Internet. HTTPS puede utilizarse, entre otros, para obtener recursos de la API web tales como datos de importación o exportación definidos en las cláusulas 9.7.2 y 7.8.4.2.

HTTPS (TLS) proporciona la seguridad de la implementación TLS subyacente del **CPE**.

NOTA – Por lo general, esta seguridad no debería utilizarse para garantizar la integridad de la protección del contenido para **Clientes ECI**, aunque puede utilizarse para poner trabas a ataques DDOS y a otros intentos oportunistas de manipular a los **Clientes ECI**.

El **Anfitrión ECI** permitirá que un **Cliente ECI** con una cantidad mínima de recursos genere peticiones Get HTTP tal como se define en [b-UIT-T J Supl. 7].

Los mensajes API de la API Get HTTP(S) figuran en el Cuadro 9.4.4.6.1-1.

Cuadro 9.4.4.6.1-1 – Mensajes de la API Get HTTP

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqHttpGetFile	A	C→H	0x0	Realiza una petición Get HTTP a un URL y almacena el resultado en un fichero.
reqHttpGetData	A	C→H	0x1	Realiza una petición Get HTTP a un URL y transfiere el resultado en forma de datos al Cliente.

9.4.4.6.2 Especificaciones aplicables

NOTA – Las especificaciones que figuran a continuación no son parte esencial de la seguridad ECI como se estipula en la cláusula 9.4.4.6.1.

La implementación de los protocolos HTTP y HTTPS para implementar la API del **Cliente ECI** será conforme con HTTP1.1 [IETF RFC 7230] y [IETF RFC 7231].

La implementación de la seguridad en la capa de transporte (TLS) utilizada para proporcionar servicios HTTP al **Cliente ECI** cumplirá con TLS 1.3 [IETF RFC 8446]. Para la compatibilidad retroactiva, se debe poder utilizar el TLS1.2 de acuerdo con las restricciones de TLS1.3 y las siguientes reglas:

1) TLS 1.2, véase [IETF RFC 5246].
2) TLS AES-GCM, véase [IETF RFC 5288].
3) TLS Extensions, véase [IETF RFC 6066].
4) PKIX/X.509 [IETF RFC 5280] + actualizaciones [IETF RFC 6818].

Todas las implementaciones TLS1.2 soportarán los conjuntos de cifrado siguientes definidos en [IETF RFC 5246]:

1) TLS_RSA_WITH_AES_128_CBC_SHA256.
2) TLS_DHE_RSA_WITH_AES_128_GCM_SHA256.

También pueden soportarse conjuntos de cifrado adicionales para TLS1.2 con las siguientes restricciones de TLS1.3.

La selección de conjuntos de cifrado TLS1.2 debe respetar las normas siguientes:

1) El conjunto de cifrado por defecto debería ser TLS_DHE_RSA_WITH_AES_128_GCM_SHA256.
2) Deberían priorizarse los conjuntos de cifrado AEAD.
3) Debería priorizarse el intercambio de claves basado en DHE.

4) No deberían priorizarse claves de una longitud superior a 128 bits.

5) No debería utilizarse 3DES.

6) No se utilizará RC4 (como se especifica en [W3C PNG]).

7) No se utilizará MD5 (como se especifica en [IETF RFC 6151]).

Son de aplicación las normas de procedimiento siguientes:

1) TLS 1.2 será la versión mínima requerida por todas las entidades **ECI**.

2) No se utilizarán SSL 2.0 y 3.0.

3) No se utilizará la renegociación.

4) No debería utilizarse la compresión (aceptable con GCM).

5) Los valores primos para DH/DHE tendrán al menos 1 024 bit y se verificarán durante los intercambios TLS.

6) La verificación de **Certificados** y anfitriones cumplirá los requisitos PKIX [IETF RFC 5280] y [IETF RFC 6125].

Los certificados raíz utilizados para autenticar la contraparte de la conexión TLS deben basarse en una lista actualizada, por ejemplo, https://cabforum.org/browser-os-info/.

Los **CPE** habrán de dar soporte a un mecanismo mediante el cual el **fabricante de CPE** pueda eliminar o desestimar certificados raíz después de la fabricación. Esto puede gestionarse a través de un mecanismo de actualización de firmware o, preferiblemente, mediante un mecanismo específico de actualización del certificado raíz que permita actualizaciones más oportunas. El **fabricante de CPE** puede optar por eliminar o desestimar un certificado raíz obligatorio en la **CPE** en respuesta a una amenaza de seguridad. Los **CPE** deben disponer de un mecanismo para añadir de forma segura nuevos certificados raíz después de la fabricación para mantener la interoperabilidad con los servidores a lo largo del tiempo.

En las normas de procesamiento identificadas por el CA/Browser Forum [b-CA Browser] y por [b-NIST SP 800-52r2] pueden encontrarse directrices adicionales para las implementaciones.

NOTA – A fin de garantizar la interoperabilidad, los servidores HTTP destinados a prestar servicios HTTP a **Clientes ECI** deberían soportar modos y opciones compatibles así como recomendaciones aplicables como las aquí definidas aquí para el cliente HTTP.

9.4.4.6.3 Mensajes reqHttpGetFile y 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[])**

- Este **mensaje** solicita al **Anfitrión ECI** que realice una petición HTTP para obtener un fichero y una vez finalizada devuelve el estado HTTP.
- resHttpGetFile devuelve el recurso como un fichero del sistema de ficheros del Cliente.
- resHttpGetData devuelve el recurso como datos de mensaje con un tamaño limitado.

Definición de los parámetros de la Petición:

fname: fileName	Nombre del fichero (filename) en el que el **Anfitrión ECI** almacena el resultado (post data) de la petición. Todos los datos existentes se sobrescriben.
url: char[]	URL codificado en UTF-8 [IETF RFC 7230]. Pueden especificarse números de puertos no normalizados formando parte del URL. Puede utilizarse TLS para los URL siempre que se cumpla el "https URI Scheme" de [IETF RFC 7230].
userAgent: char[]	Especifica el campo encabezamiento de Usuario-Agente a utilizar como encabezamiento HTTP. Los **Clientes ECI** pueden especificar un valor específico anticipado por el servidor HTTP de **url** (véase la Nota).
redirs: unit	Número máximo de redirecciones permitidas para completar la petición. Las cifras mínimas relativas a la calidad de funcionamiento de **redirs** se definen en [b-UIT-T J Supl. 7].
timeout: unit	Temporización en milisegundos para completar la petición HTTP. Si vence la temporización, la petición queda anulada y se devuelve un error de temporización en la **Contestación**.
NOTA – No es recomendable utilizar el Agente-Usuario como mecanismo de control de acceso o de selección para el recurso sino seguir el uso previsto definido en [IETF RFC 7231].	

Definición de los parámetros de la Contestación:

httpStatus: uint	Valor del estado HTTP.
data: byte[]	Datos del resultado de GET HTTP en el orden de la red. El tamaño máximo está limitado por el tamaño de la memoria intermedia del mensaje.

Información de la semántica:

- El **Anfitrión ECI** garantizará que las peticiones HTTP soportan una amplia gama de tipos comunes de ficheros y de medios. Se recomienda no incluir el campo de encabezamiento Accept en el encabezamiento de la petición HTTP. Si se añade un encabezamiento Accept podrán utilizarse los siguientes tipos MIME de codificación de contenido para obtener el recurso: application/octet-stream, application/json, image/jpeg, image/png, image/gif, text/plain, text/html, text/css, text/xml y text/javascript.
- El **Anfitrión ECI** garantizará que el Accept-Encoding del encabezamiento de la petición HTTP señalice que son aceptables las siguientes codificaciones de contenido: gzip.

Postcondiciones a la Contestación:

1) El recurso en el **url** ha sido recuperado y almacenado en un fichero de nombre **fname** (para **resHttpGetFile)** o ha sido devuelto como datos (para **rerHttpGetData**) o bien se ha producido un error.

Los códigos de error relativos a resHttpGetFile y resHttpGetData figuran en el Cuadro 9.4.4.6.3-1.

Cuadro 9.4.4.6.3-1 – Códigos de error de resHttpGetFile y resHttpGetData

Nombre	Descripción
ErrHttpGetNoSockets	Véase el Cuadro 9.4.4.6.4-1.
ErrHttpGetProtNotAvail	
ErrHttpGetPortNotAvail	
ErrHttpHostUnknown	
ErrHttpDnsOffline	
ErrHttpIpOffline	
ErrHttpTimeout	
ErrHttpGetFSFailure	
ErrHttpGetFSExceeded	
ErrHttpGetTlsAuth	
ErrHttpGetRedir	
ErrHttpGetData	

9.4.4.6.4 Códigos de error de la API Get HTTP

Los valores y significados de los errores específicos de la API que pueden devolverse en los mensajes **Contestación** de esta API figuran en el Cuadro 9.4.4.6.4-1.

Cuadro 9.4.4.6.4-1 – Códigos de error de las API Get HTTP

Nombre	Valor	Descripción
ErrHttpGetNoSockets	-257	Véanse los valores correspondientes de códigos de error en el Cuadro 9.4.4.7-1 para la API de Conector IP.
ErrHttpGetProtNotAvail	-258	
ErrHttpGetPortNotAvail	-259	
ErrHttpHostUnknown	-261	
ErrHttpDnsOffline	-263	
ErrHttpIpOffline	-267	
ErrHttpTimeout	-270	La petición HTTP no pudo finalizarse en el plazo de tiempo fijado en la petición.
ErrHttpGetFSFailure	-512	El valor +256 corresponde al valor de los códigos de error que figura en el Cuadro 9.4.5.5-1 para la API del sistema de ficheros.
ErrHttpGetFSExceeded	-514	
ErrHttpGetTlsAuth	-768	El protocolo TLS no ha podido autenticar satisfactoriamente el servidor o los datos.
ErrHttpGetRedir	-784	Se ha excedido el número de redirecciones.
ErrHttpError	-785	No ha podido obtenerse el recurso del servidor; el código de error HTTP indica el motivo.
ErrHttpGetData	-786	Los datos del recurso han superado la longitud máxima del campo de datos.

9.4.4.7 Códigos de error de la API del Conector IP

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.4.4.7-1.

Cuadro 9.4.4.7-1 – Códigos de error de la API Conector IP

Nombre	Valor	Descripción
ErrIpSourceProt	-256	Combinación inválida de fuente y protocolo.
ErrIpNoSockets	-257	No hay más conectores disponibles.
ErrIpProtNotAvail	-258	Protocolo no disponible.
ErrIpPortNotAvail	-259	El puerto solicitado no está disponible.
ErrIpSocketNotOpen	-260	El conector no estaba abierto.
ErrIpHostUnknown	-261	**Anfitrión ECI** desconocido.
ErrIpHost	-262	**Anfitrión ECI** conocido pero no existe una dirección disponible (para el tipo de dirección IP especificado).
ErrDnsOffline	-263	El servicio DNS no está en línea, posiblemente de forma temporal.
ErrIpUdpProtMismatch	-264	La dirección del par no concuerda con el protocolo del conector.
ErrIpUdpSockNot	-265	El conector no es un conector UDP.
ErrIpUdpTooLong	-266	Datagrama excesivamente largo para un único mensaje UDP.
ErrIpUdpIpOffline	-267	Conexión IP fuera de línea (no puede establecerse la conexión con el par).
ErrIpTcpProtMismatch	-268	La dirección del par no concuerda con el protocolo del conector.
ErrIpTcpSockNot	-269	El conector no es un conector TCP.
ErrIpTcpIpOffline	-258	No existe una conexión local IP a Internet en este momento.
ErrIpTcpConnRefused	-259	El anfitrión par no ha aceptado una conexión en este puerto.
ErrIpTcpConnTimeout	-260	No puede obtenerse un **Contestación** del **Anfitrión ECI** par.
ErrIpTcpClosed	-261	La conexión TCP no está o ha dejado de estar disponible.
ErrIpTcpListSockNot	-262	El conector no es un conector de servidor TCP.
ErrIpTcpNoMoreSockets	-263	La **Petición** de conexión entrante se ha recibido pero el anfitrión no tiene conectores disponibles.
RFU	Otros	Reservado para uso futuro.

9.4.5 API de acceso al sistema de ficheros

9.4.5.1 Introducción

El **Cliente ECI** tiene acceso a un sistema de ficheros privado para almacenar una cantidad limitada de datos que, en condiciones normales de funcionamiento, seguirán existiendo tras los ciclos de vida del **Cliente ECI**, los ciclos de activación o desactivación de las energías del **CPE**, las averías del sistema, etc. La fiabilidad debe ser al menos igual a la de un sistema de ficheros del **CPE** ordinario; es decir, las fallas pueden ocurrir en circunstancias en cierto modo excepcionales y que puedan dar lugar a una situación incómoda para el **Usuario**. Es labor del sistema de seguridad gestionar el **Cliente ECI** para garantizar que el **Usuario** no pierda injustificadamente derechos de acceso a contenidos. El sistema de ficheros no es seguro. En condiciones normales (es decir, cuando el **CPE** y el **Anfitrión ECI** no están comprometidos) no será posible la manipulación por parte de entidades distintas al **Cliente ECI** designado y su **Anfitrión ECI** de apoyo.

La abstracción del sistema de ficheros es la de un único directorio plano. Hay disponible un sistema de directorio básico. Las funciones de acceso al sistema de ficheros son análogas a las llamadas al sistema de ficheros Unix/Linux/Posix, como por ejemplo, open, close, write, read, lseek, opendir, readdir y lstat.

En caso de almacenamiento por el **Usuario**, cada **Cliente ECI** dispondrá de una cantidad mínima de almacenamiento en el sistema de ficheros. Esta cantidad se propone en [b-UIT-T J Supl. 7].

La API del sistema de ficheros se divide en tres partes:

1) Apertura y cierre de ficheros.
2) Lectura y escritura de un fichero, acceso aleatorio y eliminación selectiva de datos de un fichero.
3) Servicios de directorio.

Los nombres de los ficheros (Filename) constarán de una secuencia de caracteres ASCII de 8 bits con un mínimo de 1 y un máximo de 8 de los siguientes caracteres (separados por comas): A-Z, a-z, 0-9, terminados por un carácter NULO. La definición del nombre del dichero se presenta en el Cuadro 9.4.5.1-1.

Cuadro 9.4.5.1-1 – Estructura de FileName

```
typedef char fileName[9];
```

Los ficheros de registro (log) permiten a los **Clientes ECI** la escritura de cantidades limitadas de datos utilizando una memoria intermedia, es decir, sin detener la ejecución. El número de ficheros de registro por **Cliente ECI** se define en xxx (con un mínimo de 2 por cliente). Esto hace que esos ficheros sean adecuados para el registro, seguimiento y análisis post mortem a nivel de aplicación.

9.4.5.2 Apertura y cierre de ficheros

9.4.5.2.1 Generalidades

Los **Clientes ECI** pueden abrir un fichero de lectura y/o escritura, lo cual suministra un asa de fichero (fileHandle) que permite realizar los ulteriores accesos de lectura y escritura. Si un fichero no existe, puede ser creado. La característica "ubicación de fichero" apunta a su ubicación actual para facilitar el acceso al mismo.

El **Anfitrión ECI** gestionará las asas de ficheros (fileHandle). Un asa de fichero que haya sido cerrada no se reutilizará con carácter inmediato al objeto de garantizar que accesos asíncronos al fichero por parte un **Cliente ECI** no produzcan accesos a un fichero equivocado.

En el Cuadro 9.4.5.2.1-1 se definen los mensajes de apertura y cierre de ficheros:

Cuadro 9.4.5.2.1-1 – Mensajes de apertura y cierre de ficheros

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqFileOpen	A	C→H	0x0	Abre un fichero privado del **Cliente ECI**.
reqFileClose	A	C→H	0x1	Cierra un fichero abierto.

9.4.5.2.2 Mensaje reqFileOpen

C→H reqFileOpen(fileName **fname**, uint **fileOpenOptions) →**
H→C resFileOpen(uchar **fileHandle)**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** la apertura de un fichero con determinados permisos de acceso.

Definición de los parámetros de la Petición:

fname: filename	Nombre del fichero a abrir.
fileOpenOptions: unit	Modo de acceso en el que abrir el fichero. Los valores permitidos y su significado se definen en el Cuadro 9.4.5.2.2-1.

Cuadro 9.4.5.2.2-1 – Opciones de apertura de ficheros

Nombre	Bits	Valor	Descripción
FileRead	0,1	0b00	El fichero queda abierto para lectura. La ubicación del fichero se fija al inicio del mismo.
FileWriteAppend	0,1	0b01	El fichero queda abierto para escritura; las escrituras subsiguientes se añaden al fichero existente. La ubicación del fichero se fija al final del mismo.
FileWriteOver	0,1	b11	El fichero queda abierto para la escritura en cualquier ubicación. La ubicación del fichero se fija al final del mismo.
Not in use	0,1	0b10	No permitido.
LogFileNo	2	0b0	Fichero ordinario
LogFileYes	2	0b1	Fichero especial de registro que permite escritura síncrona.
Bits32-2		Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

fileHandle: uchar	Referencia (asa) a un fichero abierto.

Postcondiciones a la Petición:

1) El fichero se ha abierto en el modo de acceso deseado o bien se devuelve un error. Los códigos de error figuran en el Cuadro 9.4.5.2.2-2.

Cuadro 9.4.5.2.2-2 – Códigos de error de resfileOpen

Nombre	Descripción
ErrFileNameNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileQuotaExceeded	
ErrFileSystemFailure	

9.4.5.2.3 Mensaje reqFileClose

C→H reqFileClose(uchar **fileHandle) →**
H→C resFileClose()

1) Este mensaje cierra el acceso a un fichero abierto con **fileHandle**. Los códigos de error relativos al cierre de un fichero figuran en el Cuadro 9.4.5.2.3-1.

Definición de los parámetros de la Petición:

fileHandle: uchar	Asa del fichero a cerrar.

Precondiciones a la Petición:

1) El estado del asa de fichero (fileHandle) es abierto.

Condiciones posteriores a la Petición:

1) Los accesos subsiguientes a fileHandle serán fallidos con ErrFileNotOpen.
2) Cualquier escritura pendiente quedará comprometida (salvo que se produzca un error).

Cuadro 9.4.5.2.3-1 – Códigos de error de resfileClose

Nombre	Descripción
ErrFileHandleNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3 Acceso a ficheros

9.4.5.3.1 Generalidades

Los mensajes de acceso a ficheros permiten leer y escribir en un fichero al que se accede mediante un asa de fichero y reposicionamiento de la ubicación actual en el fichero para lectura/escritura. Las primitivas definidas tienen una correspondencia directa con convenios Linux/Unix. Los mensajes definidos figuran en el Cuadro 9.4.5.3.1-1.

NOTA – reqFileWrite y reqFileRead son muy parecidas a reqTcpSend y reqTcpRecv.

Cuadro 9.4.5.3.1-1 – Mensajes de acceso a ficheros

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqFileWrite	A	C→H	0x2	Escribe bytes consecutivos comenzando desde la ubicación actual del fichero.
reqFileRead	A	C→H	0x3	Lee bytes consecutivos comenzando desde la ubicación actual del fichero.
reqFileSeek	A	C→H	0x4	Reposiciona la ubicación actual del fichero.
reqFileRemoveData	A	C→H	0x5	Elimina datos de un fichero en su ubicación actual.
callFileDataLog	S	C→H	0x6	Añade datos al final de un fichero almacenado.

9.4.5.3.2 Mensaje reqFileWrite

C→H reqFileWrite(uchar **fileHandle**, bool **sync**, uint **dataLen**, byte **data[]) →**
H→C resFileWrite(uchar **fileHandle)**

- Este mensaje escribe bytes de dataLen en el fichero, comenzando en la ubicación actual del fichero.

Definición de los parámetros de la Petición:

fileHandle: uchar	Asa del fichero sobre el que hay que escribir
sync: booleano	Si es Verdadero, la **Contestación** de escritura garantiza que el estado del sistema de ficheros está actualizado con esta y todas las escrituras precedentes. Si es Falso, el **Anfitrión ECI** puede almacenar en la memoria intermedia **Peticiones** de escritura (que no obstante pueden perderse como consecuencia de un fallo del sistema).
dataLen: uint	Número de bytes a escribir en el fichero.
data: byte[]	Datos a escribir en el fichero.

Definición de los parámetros de la Contestación:

fileHandle: uchar	Asa del fichero en el que se realizó la escritura.

Precondiciones a la Petición:

1) El fichero está abierto en modo escritura (modo FileWriteOver o FileWriteAppend).
2) Ubicación del fichero donde puede escribirse: si el fichero está abierto en modo FileWriteAppend el fichero se ubicará al final.
3) La cantidad de datos a escribir no causa un problema de cuotas al sistema de ficheros.

Postcondiciones a la Petición:

1) El estado del fichero se actualizará y su ubicación pasará de la actual (a la espera de otras operaciones pendientes sobre el fichero almacenadas en memoria intermedia) a actual+dataLen, salvo que se produzca un error.
2) En caso de escritura y sincronización (**sync**) exitosas, los datos quedan en el estado NV en el sistema de ficheros del **Anfitrión ECI**.

Los códigos de error figuran en el Cuadro 9.4.5.3.2-1.

Cuadro 9.4.5.3.2-1 – Códigos de error de resFileWrite

Nombre	Descripción
ErrFileHandleNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.3 Mensaje reqFileRead

C→H reqFileRead(uchar **fileHandle**, uint **dataLen)** →
H→C resFileRead(uchar **fileHandle**, uint **dataRead**, byte **data[])**

- Este mensaje lee el máximo número de bytes dataLen del fichero comenzando en la ubicación actual en el fichero. Los códigos de error relacionados con la lectura de datos de un fichero figuran en el Cuadro 9.4.5.3.3-1.

Definición de los parámetros de la Petición:

fileHandle: uchar	Asa del fichero a leer.
dataLen: uint	Número máximo de bytes a leer.

Definición de los parámetros de la Contestación:

fileHandle: uchar	Asa del fichero leído.
dataRead: uint	Número de bytes leídos y almacenados en **data**.
data: byte []	Datos leídos.

Precondiciones a la Petición:

1) El fichero está abierto.

Postcondiciones a la Petición:

1) se ha producido un error; o
2) se han leído del fichero un mínimo de **dataLen** bytes o los bytes restantes desde la última posición del fichero; y
3) la posición del fichero se ha visto incrementada por el valor de **dataRead**;
4) salvo que se produzca un error, la ubicación del fichero avanzará lo indicado por **dataLen** o bien se ubicará al final del fichero.

Cuadro 9.4.5.3.3-1 – Códigos de error de resFileReade

Nombre	Descripción
ErrFileHandleNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3.4 Mensaje reqFileSeek

C→H reqFileSeek(uchar **fileHandle**, int **offset**, uchar **seekPos) →**
H→C resFileSeek(uchar **fileHandle**, int **remOffset)**

- Este mensaje coloca un puntero en determinada posición en un fichero abierto y devuelve partes del contenido del fichero.

Definición de los parámetros de la Petición:

fileHandle: uchar	Asa del fichero del que debe modificarse la posición del fichero.
offset: int	Desplazamiento con respecto a la posición de referencia de búsqueda especificada por **seekPos** que asumirá la posición en el fichero.
seekPos: uchar	Véase el Cuadro 9.4.5.3.4-1.

Cuadro 9.4.5.3.4-1 – Ubicación de referencia de búsqueda del fichero

Nombre	Valor	Descripción
FileSeekSet	0x00	La posición de referencia del fichero se encuentra al comienzo del fichero.
FileSeekCur	0x01	La posición de referencia del fichero se encuentra en la posición actual del fichero.
FileSeekEnd	0x02	La posición de referencia del fichero se encuentra al final del fichero.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

fileHandle: uchar	Asa del fichero en el que se ha modificado la posición del fichero.
remOffset: int	Diferencia entre el desplazamiento especificado y el desplazamiento para el que se fija la posición del fichero.

Información de la Semántica:

- La ubicación del fichero se reposiciona y se define en la descripción de parámetros de la **Petición**. La posición del fichero nunca estará situada más allá del final del fichero o antes del comienzo del mismo. La diferencia entre el desplazamiento solicitado y el desplazamiento real con respecto a la posición de referencia del fichero se devuelve en el parámetro del resultado **remOffset**. Los códigos de error figuran en el Cuadro 9.4.5.3.4-2.

Precondiciones a la Petición:

1) El fichero está abierto.

Postcondiciones a la Petición:

1) se ha producido un error; o

2) la posición del fichero se fija tal como se ha definido anteriormente; y

3) **remOffset** reflejará la diferencia entre el desplazamiento y la posición real del fichero tal como se define más arriba.

Cuadro 9.4.5.3.4-2 – Códigos de error de resFileReade

Nombre	Descripción
ErrFileHandleNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.3.5 Mensaje reqFileRemoveData

C→H reqFileRemoveData(uchar **fileHandle**, bool **sync**, uint **dataLen) →**
H→C resFileRemoveData(uchar **fileHandle)**

- Este mensaje elimina dataLen bytes del fichero comenzando en la ubicación actual del mismo.

Definición de los parámetros de la Petición:

fileHandle: uchar	Asa del fichero.
sync: booleano	Si es Verdadero la **Contestación** a la escritura garantiza que el estado del sistema de ficheros está actualizado con esta y todo lo escrito previamente. Si es Falso, el **Anfitrión ECI** puede almacenar en una memoria intermedia las **Peticiones** de escritura (que no obstante pueden perderse si se produce un fallo del sistema).
dataLen: uint	Número de bytes a eliminar del fichero. Si ello supera el final del fichero, solo se eliminan los bytes hasta el final del fichero.

Definición de los parámetros de la Contestación:

fileHandle: uchar	Asa del fichero en el que se ha realizado la escritura.

Precondiciones a la Petición:

1) El fichero está abierto en modo escritura (modo FileWriteOver).

Postcondiciones a la Petición:

1) El estado del fichero será actualizado. La posición del fichero seguirá siendo la misma.

2) En caso de una eliminación y sincronización (**sync)** satisfactorios, los datos quedan en estado NV en el sistema de ficheros del **Anfitrión ECI**.

Los códigos de error figuran en el Cuadro 9.4.5.3.5-1.

Cuadro 9.4.5.3.5-1 – Códigos de error de resFileWrite

Nombre	Descripción
ErrFileHandleNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.6 Mensaje callFileDataLog

C→H callFileDataLog(uchar **fileHandle**, uint **dataLen**, byte **data[])**

- Este mensaje añade dataLen bytes (en datos) al final del fichero utilizando una memoria intermedia del sistema.

Definiciones de los parámetros de la Llamada:

fileHandle: uchar	Asa del fichero.
dataLen: uint	Número de bytes a añadir al fichero registro (logfile).
data[]: byte	Datos a escribir.

Precondiciones a la Llamada:

1) El fichero está abierto en modo escritura (modo FileWriteOver o FileWriteAppend).
2) La ubicación del fichero se sitúa al final del fichero.
3) La cantidad de datos a escribir no genera un problema de cuota en el sistema de ficheros.

Postcondiciones a la Llamada:

1) El estado del fichero está actualizado y la ubicación del fichero pasa de la actual a la actual +dataLen, salvo que se produzca un error.
2) El resultado queda asignado al sistema de ficheros del **Anfitrión ECI** salvo que se produzca un error.

Información de la Semántica:

1) El **Anfitrión ECI** almacenará los datos en memoria intermedia y los añadirá al final del fichero tan pronto como convenga.
2) La capacidad máxima de memoria intermedia proporcionada para un registro con este propósito se propone en [b-UIT-T J Supl. 7].

Los códigos de error figuran en el Cuadro 9.4.5.3.6-1.

Cuadro 9.4.5.3.6-1 – Códigos de error de resFileLog

Nombre	Descripción
ErrFileHandleNotExist	Véanse las definiciones en el Cuadro 9.4.5.5-1.
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	
ErrFileLogNot	

9.4.5.4 Servicios de directorio

9.4.5.4.1 Generalidades

Los servicios de directorio ofrecen funciones para explorar los ficheros de **Cliente ECI** disponibles. Los ficheros se caracterizan por su nombre único y tienen atributos sobre su tamaño y hora de la última actualización. Los mensajes disponibles figuran en el Cuadro 9.4.5.4.1-1.

NOTA – el atributo hora tiene el mismo grado de integridad que el sistema de ficheros y el contenido del fichero.

Cuadro 9.4.5.4.1-1 – Mensajes del servicio de directorio de ficheros

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqFileStat	A	C→H	0x07	Devuelve el tamaño y la hora de modificación del fichero.
reqFileCreate	A	C→H	0x08	Crea un nuevo fichero.
reqFileDelete	A	C→H	0x09	Suprime un fichero.
reqFileDir	A	C→H	0x0A	Enumera los nombres de los ficheros disponibles en el sistema de ficheros de los **Clientes ECI**.

9.4.5.4.2 Mensaje reqFileStat

C→H reqFileStat(fileName **filename)** →
H→C resFileStat(uint **size**; long **mtime)**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** que obtenga el tamaño del fichero y la hora de la última modificación de un fichero almacenado.

Definición de los parámetros de la Petición:

filename: filename	Nombre del fichero del que se obtendrán las propiedades.

Definición de los parámetros de la Contestación:

size: uint	Tamaño del fichero (en bytes).
mtime: long	Hora de la última modificación sincronizada del fichero.

Precondiciones a la Petición:

1) Filename (nombre del fichero) es un fichero existente en el sistema de ficheros.

Postcondiciones a la Petición:

1) **size** y **mtime** reflejan las propiedades del fichero de nombre **filename** o bien se ha producido un error.

Los códigos de error figuran en el Cuadro 9.4.5.4.2-1.

Cuadro 9.4.5.4.2-1 – Códigos de error de resFileStat

Nombre	Descripción
ErrFileNameNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.3 Mensaje reqFileCreate

C→H reqFileCreate(fileName **filename)** →
H→C resFileCreate()

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** la creación de un nuevo fichero vacío. Se elimina cualquier fichero existente con el mismo nombre.

Definición de los parámetros de la Petición:

filename: filename	Nombre del nuevo fichero vacío que será creado.

Información de la Semántica:

- El fichero creado seguirá existiendo después de un fallo del sistema salvo que el sistema de ficheros se haya corrompido.

Postcondiciones a la Petición:

1) En el sistema de ficheros del **Cliente ECI** existe un fichero vacío con el nombre que figura en 'filename' con un sello de tiempo modificado para reflejar la hora actual, o bien se produce un error.

Los códigos de error figuran en el Cuadro 9.4.5.4.3-1.

Cuadro 9.4.5.4.3-1 – Códigos de error de resFileCreate

Nombre	Descripción
ErrFileQuotaExceeded	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.4 Mensaje reqFileDelete

C→H reqFileDelete(fileName **filename) →**
H→C resFileDelete()

- Este mensaje elimina el fichero cuyo nombre es el que figura en **filename**.

Definición de los parámetros de la Petición:

filename: fileName	Nombre del nuevo fichero vacío que será creado.

Información de la Semántica:

- El fichero eliminado dejará de existir tras una falla del sistema salvo que el sistema de ficheros se haya corrompido.

Postcondiciones a la Petición:

1) El fichero cuyo nombre figura en **filename** no existe en el sistema de ficheros.

Los códigos de error figuran en el Cuadro 9.4.5.4.4-1.

Cuadro 9.4.5.4.4-1 – Códigos de error de resFileDelete

Nombre	Descripción
ErrFileNameNotExist	Véase el Cuadro 9.4.5.5-1.
ErrFileSystemFailure	

9.4.5.4.5 Mensaje reqFileDir

C→H reqFileDir(ushort **maxNr) →**
H→C resFileDir(uint **listLen**; fileName **dirList[])**

- Este mensaje proporciona una lista de elementos max.maxNr de nombres de ficheros (filenames). El orden de la lista no está definido.

Definición de los parámetros de la Petición:

maxNr: ushort	Número máximo de nombres de ficheros (filenames) que se obtendrán.

Definición de los parámetros de la Contestación:

listLen: uint	Longitud en bytes de la lista.
dirList: filename []	Matriz de nombres de ficheros (filenames) de los ficheros disponibles para el **Cliente ECI**.

Los códigos de error figuran en el Cuadro 9.4.5.4.5-1.

Cuadro 9.4.5.4.5-1 – Códigos de error de resFileDelete

Nombre	Descripción
ErrFileSystemFailure	Véase el Cuadro 9.4.4.7-1.

9.4.5.5 Códigos de error de la API del sistema de ficheros

Los valores y significados de errores específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.4.5.5-1.

Cuadro 9.4.5.5-1 – Códigos de error de la API del sistema de ficheros

Nombre	Valor	Descripción
ErrFileSystemFailuret	-256	Sistema de ficheros corrupto o desarticulado.
ErrFileNameNotExist	-257	El nombre del fichero no existe en el sistema de ficheros.
ErrFileQuotaExceeded	-258	Se han excedido los recursos del sistema de ficheros del **Cliente ECI**.
ErrFileNameNotExists	-259	El nombre del fichero no existe en el sistema de ficheros del **Cliente ECI**.
ErrFileHandleNotExists	-260	El asa del fichero no existe (puede haber sido cerrada previamente).
ErrFileAppendNot	-261	El intento de escritura en el fichero no se hizo al final del fichero.
RFU	Otros	Reservado para uso futuro.

9.4.6 API de acceso al recurso hora/reloj

9.4.6.1 Introducción

El **Cliente ECI** tiene acceso a eventos del temporizador y a la hora del día a través de una API sencilla.

La robustez del reloj debe definirse por un régimen de robustez adecuado para todas las aplicaciones en un **Ecosistema ECI**.

- En caso de que se requiera que el **Ecosistema ECI** admita anti-retroceso del sistema de almacenamiento de archivos o expresiones de derechos dependientes del tiempo cuando se esté fuera de línea, el reloj debe ser robusto para que las operaciones en el almacenamiento local etiquetadas con un sello de tiempo derivado de este reloj estén adecuadamente protegidas contra la manipulación.

El temporizador permite generar un mensaje en algún momento futuro (con demora). El evento del temporizador puede ser anulado.

NOTA – Mediante una combinación de API de reloj y temporizador pueden crearse eventos periódicos del temporizador.

Las API del temporizador y del reloj se dividen en dos partes:

1) API del temporizador.

2) API del reloj.

9.4.6.2 API del temporizador

9.4.6.2.1 Generalidades

La API del temporizador permite a un **Cliente ECI** fijar un temporizador a la hora en la que transmitirá una **Contestación**. La implementación puede limitar el número de temporizadores pendientes de vencimiento en cualquier momento. El número mínimo de temporizadores pendientes de vencimiento que un **Anfitrión ECI** puede soportar para cada **Cliente ECI** se propone en [b-UIT-T J Supl. 7]. Los mensajes correspondientes a la API del temporizador figuran en el Cuadro 9.4.6.2.1-1.

Cuadro 9.4.6.2.1-1 – Mensajes de la API del temporizador

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqTimerEvent	A	C→H	0x0	Fija un evento futuro del temporizador.
reqTimerCancel	A	C→H	0x1	Cancela un evento del temporizador previamente fijado.

9.4.6.2.2 Mensaje reqTimerEvent

C→H reqTimerEvent(uint **timeInterval) →**
H→C resTimerEvent()

- Este mensaje fija el temporizador a un momento futuro y recibe una **Contestación** al vencer el temporizador.

Definición de los parámetros de la Petición:

timeInterval: uint	Hora futura en milisegundos.

Postcondiciones a la Petición:

- Después de timeInterval milisegundos se transmitirá resTimerEvent al **Cliente ECI** salvo que antes se reciba un reqTimerCancel.

Precondiciones a la Contestación:

- El temporizador ha vencido y no se ha recibido un reqTimerCancel para ese temporizador.

Los códigos de error figuran en el Cuadro 9.4.6.2.2-1.

Cuadro 9.4.6.2.2-1 – Códigos de error de resTimerEvent

Nombre	Descripción
ErrTimerMaxExceeded	Véase el Cuadro 9.4.6.4-1.

9.4.6.2.3 Mensaje reqTimerCancel

C→H reqTimerCancel(msgId **id) →**
H→C resTimerCancel()

- Este mensaje cancela un temporizador previamente fijado mediante un identificador de mensaje de la **Petición** original.

Definición de los parámetros de la Petición:

id: msgId	Cancela un temporizador que había sido fijado mediante un mensaje asíncrono con identificador de mensaje.

Precondiciones a la Petición:

1) Se ha devuelto un Id como consecuencia de un reqTimerEvent y el temporizador aún no ha vencido.

Postcondiciones a la Contestación:

1) El temporizador ha sido cancelado – no se enviará ningún resTimerCancel – o bien se devuelve un error.
2) Los errores TimerExpired se producirán si el temporizador ha sido cancelado pero se ha recibido **resTimerEvent** antes que **resTimerCancel**.

9.4.6.3 API del reloj

9.4.6.3.1 Generalidades

La API del reloj permite al **Cliente ECI** leer el reloj como un valor entero y convertirlo en una representación de la hora local. Los mensajes de la API del reloj figuran en Cuadro 9.4.6.3.1-1.

Cuadro 9.4.6.3.1-1 – Mensajes de la API del reloj

Mensaje	Tipo	Dir.	Etiqueta	Descripción
getTime	S	C→H	0x3	Lee el reloj del sistema local como un valor entero.
callLocaltime	S	C→H	0x4	Convierte el valor entero de la hora en la hora local.

9.4.6.3.2 Mensaje getTime

C→H long **getTime()**

- Este mensaje devuelve la hora, expresada en segundos, desde el 1 de enero de 1970 a las 0.00 GMT.

9.4.6.3.3 Mensaje callLocaltime

C→H callLocaltime(long **time**; tm ***tim)**

- Este mensaje convierte la **hora** en una representación humana, que se define en la estructura **tim**. Es análoga a la hora local de la función c-library tomada de <time.h>.

Definición de los parámetros de la llamada:

time: long	Hora expresada en segundos desde el 1 de enero de 1970 a las 0.00 GMT a convertir en la hora local.
tim: tm *	Puntero a la estructura tm que tomará el valor de la hora local. **tm** se define en el Cuadro 9.4.6.3.3-1.

Cuadro 9.4.6.3.3-1 – Definición de tipos de la estructura tm de representación de la hora en términos humanos

```
typedef struct tm {
   int tm_sec;  // 0 .. 59 (segundos) o 60 en caso de un segundo intercalar
   int tm_min;  // 0 .. 59 (minutos)
   int tm_hour     // 0 .. 23 (horas)
   int tm_mday; // 1 .. 31 (día del mes)
   int tm_mon;     // 1 .. 12 (mes)
   int tm_year; // year - 1900
   int tm_wday; // 0 .. 6 (día de la semana; 0=domingo)
   int tm_yday; // 0 .. 365 (día del año, 0= 1ene)
   int tm_isdst;   // 1= horario de verano en vigor, 0= no aplica horario de
                        verano
   char tm_zone[15]; // cadena de la zona horaria: p.ej. GMT, CET
   int tm_gmtoff;  // desplazamiento de la hora local respecto a GMT
} tm ;
```

9.4.6.4 Códigos de error para la API de la hora y del reloj

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.4.6.4-1.

Cuadro 9.4.6.4-1 – Códigos de error de la API de la hora y el reloj

Nombre	Valor	Descripción
ErrTimerMaxExceeded	256	Se ha excedido la máxima duración del temporizador.
RFU	Otros	Reservado para uso futuro.

9.4.7 API de acceso a la gestión energética

9.4.7.1 Introducción

El **Cliente ECI** tiene acceso a la interfaz de gestión de la energía del **Anfitrión ECI**. Esta interfaz permite al **Cliente ECI** realizar un apagado sencillo o un apagado negociado durante un evento de reposo del sistema, y permite al **Cliente ECI** reiniciar el **CPE** y al propio **Cliente ECI** en un momento posterior desde un estado energético de reposo a fin de realizar funciones de soporte. El **Anfitrión ECI** tiene los siguientes estados en relación con la alimentación de energía:

- Encendido (**PwrOn):** el **Anfitrión ECI** está en un estado operativo y no prevé realizar un apagado.
- De encendido a reposo (**PwrToStby**): el **Anfitrión ECI** desea pasar al estado de reposo (pero de forma que pueda volver al estado encendido (PowerOn)). Normalmente se solicita el apagado a todos los **Clientes ECI**.
- Reposo (**Standby**): el **Anfitrión ECI** y el **Cliente ECI** no están operativos. El **CPE** (y por tanto el **Anfitrión ECI** y el **Cliente ECI**) puede despertar de este estado por eventos previamente acordados (normalmente por un temporizador).
- Apagado (**Power-off**): el **CPE** no está alimentado. El **Anfitrión ECI** y el **Cliente ECI** no están operativos.

Los **Clientes ECI** pueden trabajar en un modo sencillo de gestión de la energía y ser apagados cómo y cuándo lo considere adecuado el **Anfitrión ECI**. Alternativamente, los **Clientes ECI** pueden solicitar permanecer en un modo gestionado mediante el envío de un mensaje **reqPwrInfo(PwrInfoOn)**. En este modo se les notificará la intención de apagado del **Anfitrión ECI** mediante el mensaje **reqPwrChange**, del que el **Cliente ECI** acusa recibo mediante **resPwrChange(PwrDown)** o lo pospone mediante un parámetro adecuado en **resPwrChange(PwrUp)** hasta el momento en que complete sus actividades en curso y esté listo para pasar al estado de reposo. El **Anfitrión ECI** reiterará regularmente el mensaje **reqPwrChange**.

NOTA – No existe una garantía absoluta de que el **Cliente ECI** pueda completar siempre todas sus actividades en curso (por ejemplo, en caso de fallo energético incontrolado o en una situación prolongada de espera para el paso a reposo).

La Figura 9.4.7.1-1 presenta el estado del **Anfitrión ECI** con las condiciones para las transiciones de estado y las actuaciones/mensajes a **Clientes ECI** en modo gestionado generados durante la transición.

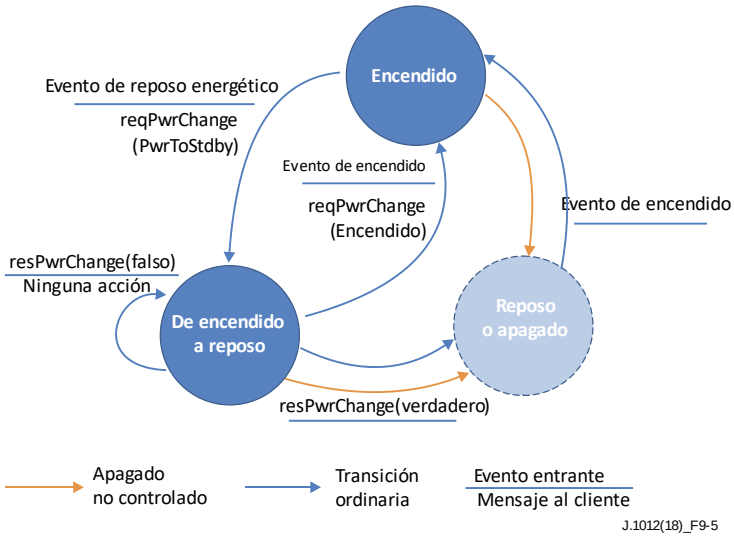


Figura 9.4.7.1-1 – Estados energéticos de un Anfitrión ECI y principales interacciones con un cliente gestionado

Los **Clientes ECI** y los **Anfitriones ECI** podrán gestionar la recuperación desde un *evento de apagado incontrolado*. En esos casos, la funcionalidad ordinaria del **Cliente ECI** y del **Anfitrión ECI** puede quedar temporalmente anulada para minimizar los problemas que pueda tener el **Usuario**.

Los **CPE** pueden tener opciones para despertar de un estado de potencia reducida basado en las características de un evento de red u otros modos de potencia reducida. La **ECI** no define un comportamiento específico para esos modos de potencia y sus interacciones con el **Anfitrión ECI** o los **Clientes ECI** distintos a los de servicios de **Anfitrión ECI** y **Cliente ECI** continuarán operativos si el estado del **Anfitrión ECI** es **PwrOn** (encendido) o **PwrToStby** (de encendido a reposo). En particular, no existe un estado específico para una ejecución suspendida.

Los **Clientes ECI** podrán posteriormente solicitar al **Anfitrión ECI** que despierte del estado de reposo y que transmita un mensaje al **Cliente ECI**.

La API de gestión energética se divide en los grupos de mensajes siguientes:

1) Transiciones energéticas: gestión ordenada del apagado de **Clientes ECI**. En la cláusula 9.4.7.2 se ofrece información adicional.

2) Funciones temporizadas de reactivación energética en nombre de los **Clientes ECI.** En la cláusula 9.4.7.3 se ofrece información adicional.

9.4.7.2 Definición de los mensajes de la API de transición energética

9.4.7.2.1 Generalidades

En esta cláusula sobre la API de gestión energética se define la funcionalidad que permite a los **Clientes ECI** realizar un apagado informado con motivo de un evento de apagado anunciado por el **Anfitrión ECI** a fin de prestar un servicio óptimo al **Usuario**. Los mensajes definidos figuran en el Cuadro 9.4.7.2.1-1.

Cuadro 9.4.7.2.1-1 – Mensajes de transición energética

Mensaje	Tipo	Dir.	Etiqueta	Descripción
getPwrStatus	S	C→H	0x0	Obtiene el estado energético actual.
setPwrInfo	S	C→H	0x1	Solicita notificaciones de eventos de cambio del estado energético.
reqPwrChange	A	H→C	0x2	Notificación de cambio del estado energético.

Los **Clientes ECI** no se consideran dados de baja tras el envío de un mensaje **resPwrInfo(PwrDown)** sino que quedan en situación de poder reiniciar sus funciones ordinarias tras la recepción de un mensaje **reqPwrChange(PwrOn)**.

9.4.7.2.2 Mensaje getPwrStatus

C→H uchar **getPwrStatus()**

- Este mensaje devuelve el estado energético actual del **Anfitrión ECI**.

Definición de propiedades: véase el Cuadro 9.4.7.2.2-1.

Cuadro 9.4.7.2.2-1 – Valores del estado energético del anfitrión

Nombre	Valor	Descripción
PwrOn	0x00	Dirección IP por defecto del **Anfitrión ECI**.
PwrToStdby	0x01	Dirección IP del **Anfitrión ECI** utilizada para las comunicaciones en la WAN (internet).
RFU	Otros	Reservado para uso futuro.

9.4.7.2.3 Mensaje setPwrInfo

C→H setPwrInfo(bool **pwrInfo)**

- Este mensaje permite entrar y salir del modo apagado gestionado, así como controlar el envío por el **Anfitrión ECI** de los mensajes **resPwrChange** al **Cliente ECI** relativos a eventos de cambio del estado energético.

Definición de propiedades:

- Si **pwrInfo** es **verdadero** se trata de un modo energético gestionado; si **pwrInfo** es **falso** el modo energético es no gestionado.

Descripción Semántica:

- Cuando **pwrInfo** es **Verdadero**, el **Anfitrión ECI** informará al **Cliente ECI** de los cambios del estado energético y no apagará el **Cliente ECI** hasta que este confirme un reqPwrChange(PwrToStby).

  Si **pwrInfo** es **Falso** el **Anfitrión ECI** no informará al **Cliente ECI** de los cambios energéticos y apagará el **Cliente ECI** "según convenga".
- Después del inicio, el estado de **PowerInfo** de cada **Cliente ECI** es **Falso**.

NOTA – Se recomienda que los **Clientes ECI** que aplican un apagado gestionado no inicien actividades sensibles al ciclo de apagado hasta que hayan enviado al **Anfitrión ECI** el mensaje **reqPwrInfo(Verdadero)**.

9.4.7.2.4 Mensaje reqPwrChange

H→C reqPwrChange(uchar **hostPwrState) →**
C→H resPowerChange(bool **ready)**

- Este mensaje señaliza un cambio del estado energético y si el argumento es **PwrToStdby** realiza una **Petición** al **Cliente ECI** para que acuse recibo o pase a reposo de forma controlada, o bien lo rechace si está realizando tares de software importantes.

Definición de los parámetros de la Petición:

hostPwrState: uchar	Nuevo estado energético del **Anfitrión ECI**. Los valores posibles se definen en el Cuadro 9.4.7.2.2-1.

Definición de los parámetros de la Contestación:

ready: booleano	Indica la preparación del **Cliente ECI** para pasar al estado de reposo.

Descripción semántica

- El **Anfitrión ECI** retransmitirá este mensaje si la **Contestación** del **Cliente ECI** es negativa (no está preparado). En [b-UIT-T J Supl. 7] se incluyen las tasas de repetición mínimas y un valor de temporización.

Precondiciones a la Petición:

1) PwrInfo == Verdadero.
2) Hubo un (reciente) cambio de estado energético en el Anfitrión ECI y el Cliente ECI (aún) no ha acusado recibo de que está listo para pasar al estado de reposo.

Postcondiciones a la Contestación:

1) El **Cliente ECI** está preparado para pasar al estado de reposo si **ready** == **Verdadero**, y no lo está si **ready == Falso**.

Los códigos de error se definen en el Cuadro 9.4.7.2.4-1.

Cuadro 9.4.7.2.4-1 – Códigos de error de ansPwrChange

Nombre	Descripción
ErrPwrInfoNot	Véase el Cuadro 9.4.7.4-1.

NOTA – Para los **Anfitriones ECI** el **error ErrPwrInfoNot** solo tiene carácter informativo.

9.4.7.3 Definición de mensajes para salir del estado de reposo

9.4.7.3.1 Generalidades

En esta cláusula relativa a la API de gestión energética se define una funcionalidad que permite a los **Clientes ECI** reanudar la ejecución a una hora previamente programada para lo que si es necesario, se despierta al CPE del estado de reposo. Los mensajes definidos figuran en el Cuadro 9.4.7.3-1.

Cuadro 9.4.7.3-1 – Mensajes para salir del estado de reposo

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setPwrWakeup	set	C→H	0x3	Fija la hora en la que el **Cliente ECI** abandonará el estado de reposo.
reqPwrWakeupEvent	A	H→C	0x4	Señaliza la expiración del temporizador despertador.

9.4.7.3.2 Mensaje setPwrWakeup

C→H setPwrWakeup(uint **time)**

- Este mensaje fija un temporizador: transcurrido **time,** el **Anfitrión ECI** despertará, si es necesario, al **Cliente ECI** del estado de reposo y transmitirá un **reqPwrWakeupEvent()**.

Definición de propiedades:

time: uint	Tiempo, en segundos, transcurrido hasta que el **Anfitrión ECI** genera un evento despertador para el **Cliente ECI**. El valor 0 significa que el **Cliente ECI** no necesita ningún evento despertador.

Información de la Semántica:

- Si nada lo impide, un **Anfitrión ECI** despertará del reposo y arrancará **inmediatamente** a un **Cliente ECI**. En caso de que tenga alguna restricción, envía el evento despertador en la primera ocasión posible. Los requisitos de precisión temporal se definen [b-UIT-T J Supl. 7].

9.4.7.3.3 Mensaje reqPwrWakeupEvent

H→C reqPwrWakeupEvent() →
C→H resWakeupEvent()

- Con este mensaje se notifica al **Cliente ECI** la expiración de su temporizador despertador a la recepción del mensaje El **Cliente ECI** acusará recibo de esta **Petición** con una **Contestación** una vez se haya completado el procesamiento crítico del evento despertador.

Información de la Semántica:

- El **Anfitrión ECI** intentará reenviar este mensaje en sucesivos eventos de inicialización del **Cliente ECI** hasta que el **Cliente ECI** acuse recibo del mismo con un mensaje **resPwrWakeupEvent()**. El evento se trasmite cuando el estado energético es encendido (**PwrOn)**, pero se demora mientras el estado es paso de encendido a reposo (**PwrToStdby**).

Precondiciones a la Petición:

1) El temporizador despertador sobre la situación energética del **Cliente ECI** se había fijado previamente y ha vencido.
2) Aún no se ha acusado recibo del evento con una **Contestación**.

3) El **Anfitrión ECI** está en el estado energético encendido (**PwrOn)**.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** detendrá el envío de mensajes **reqPwrWakeupEvent**() sobre la base del evento de cambio de estado energético de la **Petición** concordante; véase la **Precondición** 134).

9.4.7.4 Códigos de error de la API de transiciones energéticas

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.4.7.4-1.

Cuadro 9.4.7.4-1 – Códigos de error de la API de transiciones energéticas

Nombre	Valor	Descripción
ErrPwrInfoNot	-256	El **Cliente ECI** indica que no solicitó ser informado sobre eventos de cambio del estado energético.

9.4.8 API de acceso a recursos para establecer el país/idioma

9.4.8.1 Introducción

La API que relativa a los valores del país y el idioma permite a un **Cliente ECI** o a un **Anfitrión ECI** solicitar el establecimiento de los valores de país e idioma del **Usuario** de que dispone el **Anfitrión ECI** o un **Cliente ECI** respectivamente. Los mensajes de la API de establecimiento de país/idioma figuran en el Cuadro 9.4.8.1-1.

Cuadro 9.4.8.1-1 – Mensajes de la API de establecimiento de país/idioma

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqHCountry	A	C→H	0x0	Solicita el valor real preferido de país por el **Anfitrión ECI**.
reqCCountry	A	H→C	0x1	Solicita el valor real preferido de país por el **Cliente ECI**.
reqHLanguage	A	C→H	0x2	Solicita el valor real preferido de idioma del **Anfitrión ECI**.
reqCLanguage	A	H→C	0x3	Solicita el valor real preferido de idioma del **Cliente ECI**.

9.4.8.2 Definición de los mensajes de la API de país/idioma

9.4.8.2.1 Mensaje relativo al valor de país reqHCountry

C→H reqHCountry() →
H→C resHCountry setting (uint **iso_3166_country_code)**

- Este mensaje permite al **Cliente ECI** solicitar el valor del país donde actualmente reside el **Usuario,** y recibir del **Anfitrión ECI** una **Contestación** con el valor almacenado de dicho país.

Definición de parámetros de la Contestación:

iso_3166_country_code: uint	Este campo contiene el valor del país que actualmente dispone el **Anfitrión ECI**. El código de país es un campo de 24 bits que identifica el país Anfitrión mediante los tres caracteres en mayúsculas del código ISO 3166-1 alfa-3 [ISO 3166-1]. Cada carácter se codifica con 8 bits conforme a [ISO/CEI 8859-1].

Los códigos de error figuran en el Cuadro 9.4.8.2.1-1.

Cuadro 9.4.8.2.1-1 – Códigos de error de reqHCountry

Nombre	Descripción
ErrCountryNotExists	Véase el Cuadro 9.4.8.2.5-1.

9.4.8.2.2 Mensaje relativo al valor de país reqCCountry

H→C reqCCountry() →
C→H resCCountry setting (uint **iso_3166_country_code)**

- Este mensaje permite al **Anfitrión ECI** solicitar el país donde actualmente reside el **Usuario** y recibir del **Cliente ECI** una **Contestación** con el valor almacenado de dicho país.

Definición de los parámetros de la Contestación:

iso_3166_country_code: uint	Este campo contiene el valor del país que actualmente dispone el **Anfitrión ECI**. El código de país es un campo de 24 bits que identifica el país Anfitrión mediante los 3 caracteres en mayúsculas del código ISO 3166-1 alfa-3 [ISO 3166-1]. Cada carácter se codifica con 8 bits conforme a [ISO/CEI 8859-1].

Los códigos de error figuran en el Cuadro 9.4.8.2.2-1.

Cuadro 9.4.8.2.2-1 – Códigos de error de reqCCountry

Nombre	Descripción
ErrCountryNotExists	Véase el Cuadro 9.4.8.2.5-1.

9.4.8.2.3 Mensaje relativo al valor del idioma reqHLanguage

H→C reqHLanguage(uint **iso_3166_language_code) →**
C→H resHLanguage setting()

- Este mensaje permite al **Cliente ECI** solicitar el valor correspondiente al idioma actualmente preferido por el **Usuario** y recibir del **Anfitrión ECI** una **Contestación** con el valor almacenado del idioma.

Definición de los parámetros de la Contestación:

iso_3166_language_code: uint	Este campo contiene la preferencia actual del idioma del **Anfitrión ECI**. Es un campo de 24 bits que identifica el idioma mediante los 3 caracteres en minúsculas especificados en [ISO 639-2]. Pueden utilizarse tanto ISO 639-2/B como ISO 639-2/T. Cada carácter se codifica con 8 bits conforme a [ISO/CEI 8859-1].

Los códigos de error figuran en el Cuadro 9.4.8.2.3-1.

Cuadro 9.4.8.2.3-1 – Códigos de error de reqHLanguage

Nombre	Descripción
ErrLanguageNotExists	Véase el Cuadro 9.4.8.2.5-1.

9.4.8.2.4 Mensaje relativo al valor del idioma reqCLanguage

H→C reqCLanguage(uint **iso_3166_language_code) →**
C→H resCLanguage setting()

- Este mensaje permite al **Anfitrión ECI** solicitar el valor correspondiente al idioma actualmente preferido por el **Usuario** y recibir del **Cliente ECI** una **Contestación** con el valor almacenado del idioma.

Definición de los parámetros de la Contestación:

iso_3166_language_code: uint	Este campo contiene la preferencia actual del idioma del **Anfitrión ECI**. Es un campo de 24 bits que identifica el idioma mediante los 3 caracteres en minúsculas especificados en [ISO 639-2]. Pueden utilizarse tanto ISO 639-2/B como ISO 639-2/T. Cada carácter se codifica con 8 bits conforme a [ISO/CEI 8859-1].

Los códigos de error figuran en el Cuadro 9.4.8.2.4-1.

Cuadro 9.4.8.2.4-1 – Códigos de error de reqCLanguage

Nombre	Descripción
ErrLanguageNotExists	Véase el Cuadro 9.4.8.2.5-1.

9.4.8.2.5 Códigos del error de la API de establecimiento de país/idioma

Los valores de error específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.4.8.2.5-1.

Cuadro 9.4.8.2.5-1 – Códigos de error de la API de establecimiento del país/idioma

Nombre	Valor	Descripción
ErrCountryNotExists	-256	El **Anfitrión ECI** indica que el **Usuario** aún no había declarado el país donde actualmente reside.
ErrLanguageNotExists	-257	El **Anfitrión ECI** indica que el **Usuario** aún no ha declarado su idioma preferido para las comunicaciones en la interfaz de usuario.

9.5 Conjunto de las API de recursos del Anfitrión ECI específicos de la ECI

9.5.1 Lista de las API de recursos del Anfitrión ECI específicos de la ECI

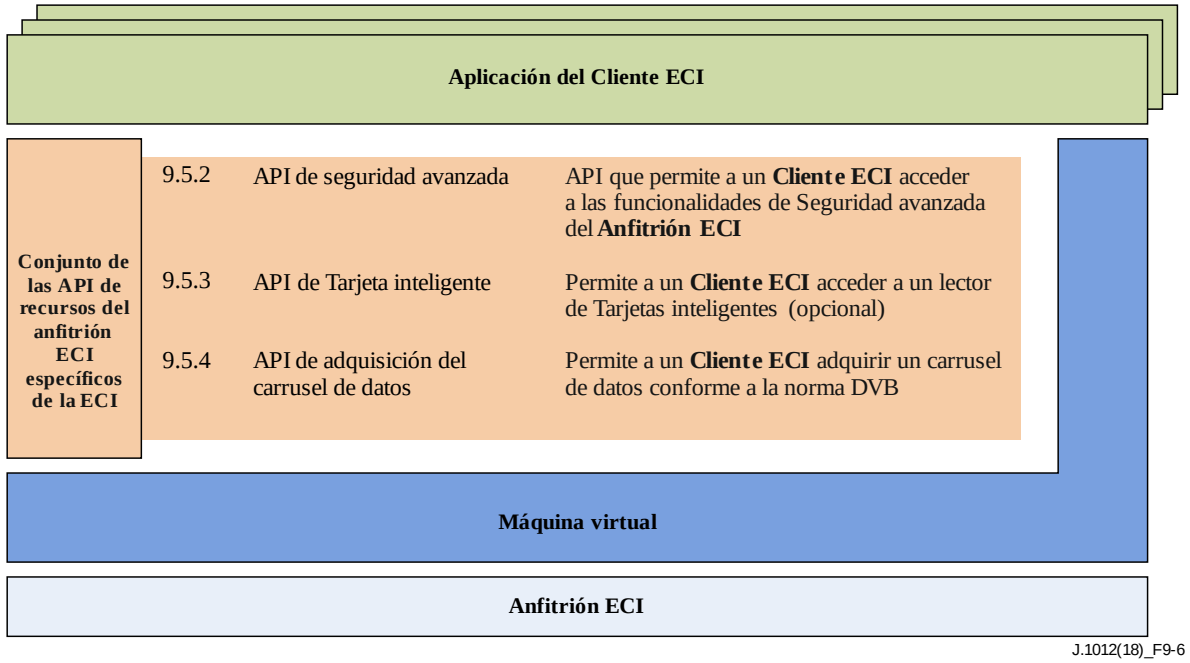


Figura 9.5.1-1 – Representación esquemática de las API definidas en la cláusula 9.5

En el Cuadro 9.5.1-1 figuran las API incluidas en la cláusula 9.5 y el Cuadro 9.5.1-1 ilustra cómo se ubican las API definidas en la cláusula 9.5 en la **arquitectura ECI**.

Cuadro 9.5.1-1 – Lista de las API definidas en la cláusula 9.5

Cláusula	Nombre de la API	Descripción
9.5.2	API de Seguridad avanzada	Permitir al **Cliente ECI** acceder a las funcionalidades de Seguridad avanzada del **Anfitrión ECI**.
9.5.3	API de **Tarjeta inteligente**	Permitir al **Cliente ECI** acceder a un lector de Tarjetas inteligentes (opcional).
9.5.4	API de Adquisición del carrusel de datos	Permitir al **Cliente ECI** adquirir un carrusel de datos conforme a la norma DVB.

9.5.2 API de Seguridad avanzada

9.5.2.1 Introducción

Cuando se carga un **Cliente ECI**, el **Anfitrión ECI** asigna un intervalo de Seguridad avanzada adecuado (un tipo de **Cliente ECI** o un tipo de **Microservidor**). Este intervalo estará disponible durante el ciclo de vida de ese **Cliente ECI**. El **Anfitrión ECI** inicializará el intervalo mediante la carga de la **Cadena de Certificados de Operación de Plataforma** que contiene la clave pública de **Operación de Plataforma.** Ello vincula cualquier intercambio significativo posterior con el intervalo AS al titular de la clave secreta de **Operación de Plataforma**.

La API de Seguridad avanzada permite a un **Cliente ECI** interactuar con la función de Seguridad avanzada (AS) en el **CPE**. Pueden producirse varios tipos de intercambios entre el **Cliente ECI** y la función AS, que normalmente inicia el **Cliente ECI**. El **Cliente ECI** recibe una señal cuando se completan operaciones AS más prolongadas.

El **intervalo AS** soporta múltiples sesiones al permitir la reutilización de la información almacenada (estado y configuración) en el **intervalo AS** para la desencriptación de varios medios y las **Sesiones de reencriptación**. El **intervalo AS** almacena para cada sesión una clave intermedia denominada "clave de enlace" de nivel superior (LK_1). Sobre la base de las LK_1 de las sesiones pueden calcularse rápidamente nuevas palabras de control para las mismas.

El **intervalo AS** también puede calcular una "Clave de autenticación" secreta que puede utilizarse en aplicaciones del **Cliente ECI**, permitiendo una entrega muy segura de información secreta al **Cliente ECI**.

El **Cliente ECI** inicializa la configuración del **intervalo AS**, que define su modo de funcionamiento. El **intervalo AS** permite al cliente autenticar su configuración; existen dos modos de autenticación fundamentales:

1) **Modo de escalera de claves**. La autenticación como parte del cálculo de la palabra de control: la configuración del intervalo se ha utilizado en el cálculo de la palabra de control con la que se ha encriptado el contenido, siendo necesaria la misma información para el cálculo de la palabra de control correcta para desencriptar el contenido, autenticando implícitamente la configuración.
2) **Modo de clave de autenticación**. La encriptación se realiza mediante una función de validación explícita que utiliza datos de verificación que sólo puede generar el provisionador del **Cliente ECI**. Esta función se requiere en la práctica para un **intervalo AS** configurado para la reencriptación ya que ésta no puede basarse en una desencriptación correcta como forma de verificación.

Además de los modos arriba indicados, el **Cliente ECI** puede solicitar una nueva verificación en la inicialización de cada intervalo, así como una "autenticación en línea". Alternativamente puede realizarse una "autenticación fuera de línea". Para lograr una autenticación satisfactoria el modo de autenticación seleccionado debe corresponder con los datos utilizados para generar la autenticación ofrecida por el provisionador.

La API de la AS en su conjunto se divide en varias API diferenciadas que reflejan las capacidades de los **Anfitriones ECI** y del **Cliente ECI** que la utilizan:

1) *API de AS general*: esta API define la funcionalidad AS genérica. Todos los **Anfitriones ECI** y los **Clientes ECI** la soportarán.
2) *API de AS de desencriptación*: esta API define la funcionalidad AS específica de desencriptación. exportación. Será soportada por todos los **Anfitriones ECI** y **Clientes ECI** con capacidades de desencriptación.

3) *API de AS de exportación*: esta API define la funcionalidad AS específica de exportación. Será soportada por todos los **Anfitriones ECI** y **Clientes ECI** con capacidades de desencriptación y exportación. Los **Anfitriones ECI** que soportan la exportación también soportarán la encriptación.

4) *API de AS de encriptación*: esta API define la funcionalidad AS específica de encriptación. Será soportada por todos los **Anfitriones ECI** y **Clientes ECI** con capacidades de encriptación.

Se aplicarán las limitaciones siguientes:

- Un **Cliente ECI** soportará la desencriptación o la encriptación, no siendo exigible que soporte ambas simultáneamente.

El **Anfitrión ECI** y el **Cliente ECI** utilizarán el recurso de descubrimiento de la interfaz del **Anfitrión ECI** para intercambiar información sobre sus respectivas capacidades. El **Anfitrión ECI** asignará el intervalo adecuado de conformidad con el resultado del descubrimiento: un intervalo de encriptación para **Clientes ECI** que requieran encriptación y un intervalo de AS de desencriptación para **Clientes ECI** que requieran desencriptación.

NOTA – Pueden existir funciones que proporcionen funcionalidades complementarias en distintas API: la API de AS general y una API de AS más específica.

Los mensajes de la API de AS general sólo requieren ser soportados por el **Anfitrión ECI** en la medida en que sea necesario para reflejar las capacidades del **Anfitrión ECI** (soporte de desencriptación, exportación y encriptación).

Los mensajes de las API de AS se definen en términos de las funciones AS definidas en las cláusulas 8.2.4 y 9.9 de [UIT-T J.1014]. La cláusula 8.2.4.1 de [UIT-T J.1014] ofrece una visión general de la función AS. El primer parámetro, slotId, se omite en las definiciones de [UIT-T J.1014]: lo suministra el **Anfitrión ECI**.

Muchas de las definiciones de tipos y valores de parámetros tal como se utilizan en esta definición de API, se establecen en [UIT-T J.1014]. Los códigos de error de esta API se definen en [UIT-T J.1014], y no figuran en esta Recomendación para cada mensaje. Los códigos de error para los valores de parámetros corresponden al cómputo de la secuencia de parámetros, tal como se define en las funciones a las que se hace referencia de [UIT-T J.1014], que normalmente tienen un parámetro adicional (slotId).

9.5.2.2 Definición de mensajes de la API de seguridad avanzada general

9.5.2.2.1 Generalidades

Los mensajes de la API de seguridad avanzada general figuran en el Cuadro 9.5.2.2.1-1.

Cuadro 9.5.2.2.1-1 – Mensajes generales de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqAsInitSlot	A	C→H	0x0	Inicializa el **intervalo AS**.
callAsNextKeySession	S	C→H	0x1	Cambia a la siguiente clave aleatoria para una sesión.
reqAsStopSession	A	C→H	0x2	Detiene una sesión.
reqAsLoadSlotLk	A	C→H	0x3	Calcula la clave del enlace de nivel superior (LK1).
reqAsComputeAkClient	A	C→H	0x4	Calcula la clave de autenticación para aplicaciones del **Cliente ECI**.
reqAsClientChalResp	A	C→H	0x5	Aplica la clave de autenticación del **Cliente ECI** a los datos y devuelve un resultado.
getAsSlotRk	S	C→H	0x6	Obtiene una clave aleatoria para el **intervalo AS**.
getAsSessionRk	S	C→H	0x7	Obtiene una clave aleatoria para una sesión.
getAsSessionLimitCounter	S	C→H	0x8	Obtiene un valor límite actual del contador de para la sesión.

Cuadro 9.5.2.2.1-1 – Mensajes generales de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setAsSessionLimitEvent	S	C→H	0x9	Fija el valor límite para enviar un mensaje reqAsEventSessionLimit al **Cliente ECI**.
reqAsEventSessionLimit	A	H→C	0xA	Al alcanzar un valor límite para las unidades restantes se envía un evento al **Cliente ECI**.
getAsClientRnd	S	C→H	0xB	Obtiene un nuevo número aleatorio para aplicaciones del **Cliente ECI**.
getAsSC	S	C→H	0xC	Obtiene el estado del campo control de aleatorización actual del contenido en una sesión.
reqAsEventSC	A	H→C	0xD	Mensaje de evento cuando cambia el campo control de aleatorización en una sesión.
getChipsetId	S	C→H	0xE	Obtiene el valor de ChipsetID del **Bloque de escalera de claves**.
getImageTargetId	S	C→H	0xF	Obtiene el valor de ECI_Image_Target_Id del CPE.

9.5.2.2.2 Mensaje reqAsInitSlot

C→H reqAsInitSlot(uint **slotVersion**, uint **slotMode)** →
H→C resAsInitSlot()

- Este mensaje inicializa el intervalo con varios parámetros generales.

Definición de los parámetros de la Petición:

slotVersion: uint	Versión de la funcionalidad del intervalo definida en [UIT-T J.1014].
slotMode: uint	Modo principal de operación del intervalo; véase [UIT-T J.1014].

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsInitSlot definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor de los parámetros slotId y POPKchain.

9.5.2.2.3 Mensaje callAsNextKeySession

C→H callAsNextKeySession(uint **sessionId)**

- Este mensaje provoca el cambio de la siguiente clave aleatoria de una sesión.

Definición de los parámetros de la Petición:

sessionId: uint	Sesión para la que se debe cambiar a la siguiente clave aleatoria.

Descripción Semántica:

- Este mensaje es equivalente al mensaje AS callAsNextKeySession definido en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.4 Mensaje reqAsStopSession

C→H reqAsStopSession(uint **sessionId)** →
H→C resAsStopSession()

- Este mensaje detiene una sesión de **intervalo de SA**.

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión a detener.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsStopSession definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.5 Mensaje reqAsLoadSlotLk

C→H reqAsLoadSlotLk(uint **sessId**, InputV, ulong **spkUri**, uchar **spkIndx) →**
H→C resAsLoadSlotLk()

- Este mensaje calcula la clave del enlace de nivel superior LK_1 que puede utilizarse posteriormente para calcular palabras de control.

Definición de los parámetros de la Petición:

sessId: uint	Id de sesión a inicializar.
inputV: InputV	Mensaje que contiene la clave pública del conjunto de chips encriptada y la LK1 de la clave de enlace protegida de la firma de la clave secreta del emisor.
spkUri: ulong	Normas de utilización del vector SPK utilizado posteriormente para calcular una palabra de control, véase [UIT-T J.1014].
spkIndx: uchar	Índice que define la ubicación de la SPK del **intervalo AS** en el vector SPK utilizado posteriormente para calcular una palabra de control, véase la cláusula 7 de [UIT-T J.1014].

Descripción Semántica:

- Este mensaje es equivalente a la función de **intervalo AS** reqAsLoadSlotLk definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.
- El **Anfitrión ECI** también emitirá una función reqAsDeoupleDecryptSession [UIT-T J.1014] si se detiene una sesión de desencriptación del **intervalo AS** previamente emparejada con otra sesión de desencriptación del **intervalo AS** (véase la cláusula 9.5.2.3.1).

9.5.2.2.6 Mensaje 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()

- Este mensaje calcula una clave de autenticación para su uso por el **Cliente ECI**.

Definición de los parámetros de la Petición:

inputV: InputV	Mensaje que contiene la clave pública del conjunto de chips y el valor r protegido de la firma de la Clave secreta del emisor usado para calcular AK.
nSpk: uint	Número de valores en el vector SPK, véase [UIT-T J.1014].
spkIndx: uchar	Índice que define la ubicación del SPK del intervalo AS en el vector SPK, el valor POPK del intervalo AS en el vector POPK y el slotConfig del intervalo AS en el vector clCnf utilizado para calcular la Clave de autenticación del cliente, véase [UIT-T J.1014].
spk[16]: PubKey	Vector **Clave pública del emisor** utilizado para calcular la Clave de autenticación del cliente; véase [UIT-T J.1014].
popk[16]: PubKey	Vector Clave pública del operador de plataforma utilizado para calcular la Clave de autenticación del cliente; véase [UIT-T J.1014].
akCnf[16]: SessionConfig	Vector configuración de la sesión del cliente utilizado para calcular la Clave de autenticación del cliente; véase [UIT-T J.1014].
spkUri: ulong	Normas de utilización del vector SPK usado posteriormente para calcular una palabra de control, véase [UIT-T J.1014].
XT[32]: uchar	Valor del campo extensión usado para calcular la Clave de autenticación del cliente; véase [UIT-T J.1014]. El valor por defecto es { 0x00 }.
online: booleano	Si es verdadero, la clave aleatoria del intervalo se utiliza para calcular la Clave de autenticación que obliga al provisionador a calcular de nuevo la Clave de autenticación.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsComputeAkClient definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.7 Mensaje reqAsClientChalResp

C→H **reqAsClientChalResp(**uchar **challenge**[16]**)**; →
H→C **reqAsClientChalResp(**uchar **response**[16]**)**

- Este mensaje utiliza la Clave de autenticación de cliente, tal como es calculada mediante el mensaje reqAsComputeAkClient (definido en [UIT-T J.1014]), para desencriptar una entrada del parámetro de desafío de 128 bits y generar como resultado un parámetro de contestación de 128 bits.

Definición de los parámetros de la Petición:

challenge[16]: uchar	Entrada de 128 bits a desencriptar mediante la Clave de autenticación del cliente.

Definición de los parámetros de la Contestación:

response[16]: uchar	Salida de 128 bits desencriptados.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsClientChalResp definida en [UIT-T J.1014]; el **Anfitrión ECI** proporciona el valor del parámetro slotId y el mensaje **Contestación** transporta el resultado del parámetro "contestación".

9.5.2.2.8 Mensaje getAsSlotRk

C→H SymKey **getAsSlotRk()**

- Este mensaje lee la clave aleatoria para la sesión del **intervalo AS** del **Cliente ECI**.

Descripción Semántica:

- Este mensaje es equivalente a la función AS getAsSlotRk definido en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.9 Mensaje getAsSessionRk

C→H SymKey **getAsSessionRk(**uint **sessionId**, uint **rkIndx)**

- Este mensaje lee la clave aleatoria actual (rkIndx==0) o siguiente (rkIndx==1) para la sesión del **Cliente ECI** cuyo identificador es sessionId.

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión para la que debe obtenerse la clave de sesión aleatoria.
rkIndx: uint	Identifica si debe obtenerse la clave de sesión aleatoria actual (rkIndx==0) o siguiente (rkIndx==1).

Descripción Semántica:

- Este mensaje es equivalente al mensaje AS getAsSessionRk definido en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.10 Mensaje getAsSessionLimitCounter

C→H ulong **getAsSessionLimitCounter(**uint **sessionId)**

- Este mensaje devuelve el valor límite del contador para el sessionId del **Cliente ECI**.

Descripción Semántica:

- Esta función es equivalente a la función AS getAsSessionLimitCounter definida en [UIT-T J.1014]; el **Anfitrión ECI** proporciona el valor del parámetro slotId.

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión para la que debe obtenerse el contador de límite de sesión.

9.5.2.2.11 Mensaje setAsSessionLimitEvent

C→H ulong **setAsSessionLimitEvent(**uint **sessionId**, ulong **eventLimit)**

- Este mensaje fija el valor límite eventLimit del limitCounter de la sesión del **Cliente ECI** cuyo identificador es sessionId para un mensaje reqAsEventSessionLimit a devolver al **Cliente ECI**.

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión para la que se debe fijar el eventLimit de sesión.
eventLimit: ulong	Valor del límite de evento que debe fijarse.

Descripción Semántica:

- Esta función es equivalente a la función AS setAsSessionLimitEvent definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.12 Mensaje reqAsEventSessionLimit

H→C **reqAsEventSessionLimit(**uint **sessionId) →**
C→H res**AsEventSessionLimit()**

- Este mensaje devuelve el valor límite del contador de la sessionId del **Cliente ECI**.

Definición de los parámetros de la Contestación:

sessionId: uint	Id de la sesión en la que se ha generado un evento eventLimit.

Descripción Semántica:

- Esta función es equivalente a la función reqAsEventSessionLimit de AS definida en [UIT-T J.1014]; el **Anfitrión ECI** elimina el parámetro slotId.

9.5.2.2.13 Mensaje getAsClientRnd

C→H SymKey **getAsClientRnd()**

- Este mensaje devuelve un número aleatorio de 128 bits.

Descripción Semántica:

- Esta función es equivalente al mensaje AS getAsClientRnd definido en [UIT-T J.1014].

9.5.2.2.14 Mensaje getAsSC

C→H uint **getAsSC(**uint **sessionId)**

- Este mensaje devuelve el estado actual del campo Control de aleatorización del contenido en una sesión.

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión para la que debe obtenerse el valor actual del campo control de aleatorización.

Descripción Semántica:

- Esta función es equivalente a la función AS getAsSC definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.2.15 Mensaje reqAsEventSC

H→C reqAsEventSC(uint **sessionId**; uint **scramblingControlField)** →
C→H resAsEventSC()

- Este mensaje indica que se ha producido un cambio en el campo control de la aleatorización en la sesión cuyo identificador es sessionId.

Definición de los parámetros de la Contestación:

sessionId: uint	Id de la sesión en la que se ha producido un cambio en el campo estado de aleatorización.
scramblingControlField: uint	Nuevo valor del campo estado de aleatorización. Véase en la cláusula 9.9 de [UIT-T J.1014] la definición de los valores y su semántica.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsEventSC definida en [UIT-T J.1014]; el **Anfitrión ECI** elimina en valor del parámetro slotId.

9.5.2.2.16 Mensaje getChipsetId

C→H ulong **getChipsetId()**

- Este mensaje devuelve el valor de ChipsetID del **Bloque de escalera de claves** definido en [UIT-T J.1014].

9.5.2.2.17 Mensaje getImageTargetId message

C→H ECI_Image_Target_Id **getImageTargetId()**

- Este mensaje devuelve el valor de ECI_Image_Target_Id del CPE, definido en el Cuadro 6.2.2.2-1.

9.5.2.3 Definición de mensajes API de desencriptación de seguridad avanzada

9.5.2.3.1 Generalidades

La API de desencriptación de Seguridad avanzada proporciona los mensajes que figuran en el Cuadro 9.5.2.3.1-1.

Es posible emparejar dos sesiones de desencriptación y permitir el uso de distintas palabras de control al objeto de desencriptar dos flujos de contenido que deben tratarse como un único elemento de contenido después de la desencriptación.

EJEMPLO: Un canal de deportes se puede difundir con varios canales de audio, siendo accesible el canal de audio de un idioma dado sólo si se dispone de la suscripción específica para desencriptarlo. Una sesión sólo puede emparejarse con otra.

Cuadro 9.5.2.3.1-1 – Mensajes de desencriptación de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqAsAStartDecryptSession	A	H→C	0x0	Inicia una sesión de desencriptación en el **intervalo AS** del **Cliente ECI**.
reqAsComputeDecrCw	A	H→C	0x1	Calcula una palabra de control de desencriptación.
reqAsAuthDecrSlotConfig	A	H→C	0x2	Autentica la configuración del intervalo con mecanismos de autenticación (modo de desencriptación).

9.5.2.3.2 Mensaje reqAsStartDecryptSession

C→H reqAsAStartDecryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config,** ScrambleMode **sm)** →
H→C resAsAStartDecryptSession(uint **sessionId)**

- Este mensaje comienza una sesión de desencriptación en el **intervalo AS** del **Cliente ECI**.

Definición de los parámetros de la Petición:

mh: ushort	**Distintivo de medios** para el que se desencripta el contenido (usada por el **Anfitrión ECI** para asociar el contenido a desencriptar al recurso de desencriptación asignado a esta sesión).
spk: PubKey	Clave pública del emisor para esta sesión.
config: SessionConfig	Configuración de la sesión.
sm: ScrambleMode	Modo de desaleatorización a utilizar. Para su definición véase el Cuadro 9.5.2.3.2-1. Véase la Nota.
NOTA – La información del parámetro sm no debe entrar en contradicción con el parámetro cwUri de un mensaje reqAsComputeDecrCw posterior.	

Cuadro 9.5.2.3.2-1 – Definición de ScrambleMode

```
typedef ScrambleMode {
  uchar     modeRef;
  uchar     mode[16] ;
} ScrambleMode;
```

La definición de **modeRef** figura en el Cuadro 9.5.2.3.2-2.

Cuadro 9.5.2.3.2-2 – Definición de modeRef

Nombre	Valor	Descripción
ScrambleModeHost	0x01	El anfitrión seleccionará el modo de (des)aleatorización sobre la base de información estándar o patentada.
ScrambleModeDvb	0x02	Se utiliza la definición DVB para el modo de aleatorización. El byte 0 del campo modo contiene un valor con el mismo significado que se define en el campo scrambling_mode del Scrambling_descriptor, según se define en [CEI 62766-5-2]. El byte 1 tiene el siguiente significado cuando el valor del byte 0 es 0x02, 0x03 y 0x10 (es decir, los modos DVB CSA1/2 y DVB CSA3 para la desaleatorización y el modo DVB-CISSA versión 1): Valor==0x01: modo TS (des)aleatorización. Valor==0x02: modo PES (des)aleatorización. Todos los demás valores están reservados; todos los bytes no utilizados del campo modo están reservados. Véase la Nota 1.
ScrambleModeCencEnum	0x03	El modo de aleatorización se define en [UIT-T T.871] definiéndose el byte 0 del campo modo como: Valor==0x01: modo CENC CTR. Valor==0x02: modo CENC CBC. El resto de valores del byte 0 están reservados. Para los valores arriba definidos del byte 0, el byte 1 sigue el subesquema siguiente: Valor==0x01: anfitrión definido, para la encriptación seleccionada conforme a uno de los valores definidos a continuación. Valor==0x02: encriptación del segmento completo, tal como se define en [W3C GIF V89a]. Valor==0x03: encriptación de submuestra, tal como se define en [W3C PNG]. El resto de valores del byte 1 están reservados. Para otros valores de byte 0, el byte 1 está reservado. Los bytes 2-15 están reservados. Véase la Nota 2.
RFU	Otros	Reservado para uso futuro.
NOTA 1 – El **Anfitrión ECI** soportará al menos los modos DVB CSA1/2 y DVB CSA3 para la desaleatorización y el modo DVB-CISSA versión 1 para aleatorización y desaleatorización. NOTA 2 – El **Cliente ECI** o (si está permitido) el **Anfitrión ECI** pueden seleccionar un modo de aleatorización para la encriptación que se adapte adecuadamente a la aplicación; en particular, teniendo en cuenta que las aplicaciones del tipo flujo normalmente utilizan la encriptación de segmento completo CBC y las aplicaciones de almacenamiento normalmente utilizan el modo CTR, y pueden beneficiarse de la encriptación de submuestra.		

Definición de los parámetros de la Contestación:

sessionId: uint	Id de la sesión que ha sido creada.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsAStartDecryptSession [UIT-T J.1014]; el **Anfitrión ECI** proporciona el valor del parámetro slotId, y el resultado sessionId se devuelve en el mensaje **Contestación**.

El **Anfitrión ECI** también generará una función reqAsCoupleDecryptSession [UIT-T J.1014] cuando se inicia una segunda sesión de desencriptación de **intervalo AS** para el mismo **Asa de Medios**, de forma que se emparejan estas sesiones de desencriptación de **intervalo AS**, emparejándose la segunda sesión con la primera.

9.5.2.3.3 Mensaje 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, uint **cwIndx)** →
H→C resAsComputeDecrCw ()

- Este mensaje calcula una palabra de control de desencriptación.

Definición de los parámetros de la Petición:

sessionId: int	Id de la sesión para la que debe calcularse una palabra de control.
cwUri: ulong	El cwUri define las aplicaciones de la palabra de control. Los valores de cwUri se definen en la cláusula 7.5 de [UIT-T J.1014].
nSpk: uint	Número de valores SPK en el vector SPK.
nElk: uint	Número de valores Elk en el vector ELK.
elk[24]: SymKey	Vector de valores de claves con encriptación simétrica que se desencriptarán sucesivamente mediante el mecanismo de escalera de claves. El valor elk[nElk-2] es la entrada de datos del campo1 a la autenticación de las propiedades del contenido, tal como se define en la cláusula 8.2.3 de [UIT-T J.1014], utilizando la función definida en la cláusula 8.2.4.7 de [UIT-T J.1014].
spk[16]: PubKey	Vector de claves públicas del emisor tal como se define en [UIT-T J.1014], cláusula 5.5.
popk[16]: PubKey	Vector de claves públicas del operador de plataforma, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
config[16]: SessionConfig	Vector de configuraciones de sesiones de cliente, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
XT[32]: uchar	Entrada de datos de reserva para el mecanismo de la palabra de control, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
rkIndx: uint	Identifica si en el cálculo de la palabra de control se utiliza la clave de sesión aleatoria actual (rkIndx==0) o la siguiente (rkIndx==1).
field2: Field2	Contenido relativo a las propiedades del contenido de mayor tamaño no autenticado en el campo1, tal como se define en la cláusula 8.2.3 de [UIT-T J.1014].
cwIndx: uint	Índice de la palabra de control a calcular: 0 para la palabra de control par y 1 para la impar; carece de significado para la desencriptación basada en ficheros.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsComputeDecrCw definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.3.4 Mensaje 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 ()

- Este mensaje autentica la configuración del intervalo con mecanismos de autenticación (modo de desencriptación).

Definición de los parámetros de la Petición:

sessionId: uint	Id de la sesión para la que debe autenticarse la configuración del intervalo.
inputV: InputV	Mensaje que contiene la clave pública encriptada del conjunto de chips y el valor r protegido de la firma de la Clave secreta del emisor usado para calcular la AK utilizada a fin de autenticar la configuración del **intervalo AS**.
nSpk: uchar	Número de valores SPK en el vector SPK.
spkIndx: uint	Índice que define la ubicación del SPK del intervalo AS en el vector SPK, el valor POPK del intervalo AS en el vector POPK y el slotConfig del intervalo AS en el vector clCnf utilizado para calcular la Clave de autenticación del cliente, véase [UIT-T J.1014].
spk[16]: PubKey	Vector de claves públicas del emisor tal como se define en la cláusula 7.5 de [UIT-T J.1014].
popk[16]: PubKey	Vector de claves públicas del operador de plataforma tal como se define en la cláusula 7.5 de [UIT-T J.1014].
cnf[16]: SessionConfig	Vector de las configuraciones de cliente tal como se define en la cláusula 7.5 de [UIT-T J.1014].
spkUri: ulong	Normas de utilización del vector SPK usado posteriormente para calcular la Clave de autenticación AK, véase [UIT-T J.1014].
XT[32]: uchar	Valor del campo extensión usado para calcular la Clave de autenticación de cliente; véase [UIT-T J.1014]. El valor por defecto es { 0x00 }.
online: booleano	Si es verdadero, se utiliza la clave aleatoria del intervalo para calcular la Clave de autenticación obligando al provisionador a realizar un nuevo cálculo de la Clave de autenticación.
verifier[16]: uchar	Valor con el que reqAsAuthDecrSlotConfig autentica la configuración del intervalo.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsAuthDecrSlotConfig definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.4 API de exportación de seguridad avanzada

9.5.2.4.1 Generalidades

Los mensajes de la API de exportación de seguridad avanzada figuran en el Cuadro 9.5.2.4.1-1.

Cuadro 9.5.2.4.1-1 – Mensajes de exportación de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqAsExportConnSetup	A	C→H	0x0	Establece una **Conexión de exportación** entre la sesión de desencriptación y la de encriptación.
reqAsExportConnEnd	A	C→H	0x1	Termina la sesión de exportación en curso.

9.5.2.4.2 Mensaje reqAsExportConnSetup

C→H reqAsExportConnSetup(uint **sessId**, ushort **expMh**, uint **grpIndx**;
CertSerialChain **expCh**, CertSerialChain **impCh**, CertSerialChain **auth**[]**) →**
H→C resAsExportConnSetup()

- Este mensaje establece una conexión de seguridad avanzada desde la sesión de desencriptación a la sesión del **Asa de Medios** de exportación.

Definición de los parámetros de la Petición:

sessId: uint	Id de la sesión de exportación del **intervalo AS** del **Cliente ECI**.
expMh: ushort	Id del **Asa de Medios** de exportación que debe utilizarse para la encriptación del contenido desencriptado en las sesiones AS.
grpIndx: uint	Índice para almacenar la conexión de la sesión de exportación; los valores permitidos son 0 y 1. Este parámetro puede utilizarse como sustitutivo de la autenticación de la **Conexión de exportación** con un **Microservidor** (por ejemplo, para anticipar un próximo cambio del ID de **Grupo de exportación** en un flujo).
expCh: CertSerialChain	**Cadena de exportación** para un **Cliente ECI**.
impCh: CertSerialChain	**Cadena de importación** para la encriptación/importación de un **Cliente ECI**.
auth[]: CertSerialChain	**Certificados** de autorización para la **Cadena de importación**.

Descripción Semántica:

- Este mensaje es equivalente a la función **AS** reqAsExportConnSetup definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor de los parámetros slotId, impSlotId e ImpSessId asociados. El **Anfitrión ECI** usará el **Asa de Medios** de la sesión de exportación para conectar la sesión de desencriptación AS con la correspondiente sesión de encriptación AS, es decir, proporcionar los parámetros impSlotId e impSessId en la función AS reqAsExportConnSetup de [UIT-T J.1014].

9.5.2.4.3 Mensaje reqAsExportConnEnd

C→H reqAsExportConnEnd(ushort **expMh) →**
H→C resAsExportConnEnd()

- Este mensaje da por terminada una sesión de exportación en curso.

Definición de los parámetros de la Petición:

expMh: ushort	Exporta la sesión de **Asa de Medios** de las sesiones AS para las que se terminará el intercambio de contenido.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsExportConnEnd definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor de los parámetros slotId y sessionId asociados con expMh.

9.5.2.5 API de encriptación de seguridad avanzada

9.5.2.5.1 Generalidades

Los mensajes de la API de encriptación de seguridad avanzada figuran en el Cuadro 9.5.2.5.1-1.

Cuadro 9.5.2.5.1-1 – Mensajes de encriptación de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqAsStartEncryptSession	A	C→H	0x0	Inicia una sesión de encriptación.
reqAsComputeEncrCw	A	C→H	0x1	Calcula la palabra de control de encriptación.
reqAsAuthEncrSlotConfig	A	C→H	0x2	Autentica la configuración del intervalo y los parámetros de encriptación con mecanismos de autenticación (modo de encriptación)
reqAsLdUssk	A	C→H	0x3	Carga la clave secreta del **Microservidor**.
reqAsMInikLk1	A	C→H	0x4	Calcula el mensaje de inicialización asimétrica del **Microcliente**.
reqAsEventCpChange	A	H→C	0x5	Mensaje de evento sobre la modificación de las propiedades del contenido de un contenido importado en una sesión de encriptación

Cuadro 9.5.2.5.1-1 – Mensajes de encriptación de seguridad avanzada

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setAsPermitCPChange	S	C→H	0x6	Habilita/deshabilita cambios en la propiedad del contenido (CP) importado que se producen en una sesión de encriptación durante la selección de la palabra de control para la encriptación.
setAsSC	S	C→H	0x7	Fija el campo control de aleatorización del contenido encriptado de una sesión de encriptación.

9.5.2.5.2 Definición de la Cadena de cliente objetivo

Los **Microservidores** pueden utilizar el **Sistema de procesamiento de certificados** para proporcionar una implementación robusta de la autenticación asimétrica del cliente. La **ECI** define cadenas de certificados para permitir dicha autenticación de **Microcliente**. Esas cadenas objetivo se utilizan como entradas al mensaje reqAsMInikLk1.

Las **Cadenas de certificados** serán conformes con la cláusula 5.4.1. Se utilizan dos tipos de **Certificados**:

- **Certificado de Microcliente.** que autentica a un único **Microcliente**; la clave pública del **Certificado** será idéntica a la Clave pública del juego de circuitos del **CPE Microcliente,** en caso de que el Microcliente sea un **Cliente ECI**.
- **Certificado** de grupo objetivo, que autentica uno o más Grupos objetivo o **Certificados de Microcliente**.

Los operadores de **sistemas microDRM** pueden utilizar el mecanismo de **Lista de Revocación ECI** para gestionar de forma segura la evolución de **Microclientes** autenticados para un servidor.

NOTA – El mantenimiento de **Listas de Revocación** constituyen un asunto privado del operador del **sistema microDRM**.

El ID de **Certificado** para el **Certificado** del Grupo objetivo se define en el Cuadro 9.5.2.5.2-1.

Cuadro 9.5.2.5.2-1 – Definición del ID de Grupo objetivo

Sintaxis	N.° de bits	Mnemónico
`ECI_Target_Group_Id {`		
`padding(4)`		
`type`	4	uimsbf
`target_group_id`	20	uimsbf
`target_group_version`	8	uimsbf
`}`		

Semántica:

type: entero	Valor conforme con el Cuadro 5.1.3-1.
target_group_id: entero	Número de **Grupo Objetivo**, único en el contexto del **Padre**.
target_group_version: entero	Incrementa su valor si el microgrupo cambia su **Certificado**.

El ID de **Certificado** para el **Certificado de Microcliente** se define en el Cuadro 9.5.2.5.2-2.

Cuadro 9.5.2.5.2-2 – Definición del ID de Microcliente

Sintaxis	N.° de bits	Mnemónico
ECI_Micro_Client_Id {		
padding(4)		
type	4	uimsbf
micro_client_id	20	uimsbf
micro_client_version	8	uimsbf
}		

Semántica:

type: entero	Valor conforme con el Cuadro 5.1.3-1.
micro_client_id: entero	Número del **Microcliente**, único en el contexto del **Padre**.
micro_client_version: entero	Se ve incrementado si el microgrupo cambia su **Certificado**.

9.5.2.5.3 Mensaje reqAsStartEncryptSession

C→H reqAsStartEncryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config**, uint **nEncr**, PubKey **encrSpk[MaxSpkEncr]**, PubKey **encrPopk[MaxSpkEncr]**, ulong **encrCwUri)** →
H→C resAsStartEncryptSession()

- Este mensaje inicia la sesión de encriptación.

Definición de los parámetros de la Petición:

mh: ushort	Identificador del **Asa de Medios** del contenido encriptado para el que se debe crear una sesión de encriptación.
spk: PubKey	Clave pública el emisor utilizada por el Sistema AS para autenticar al emisor y el mensaje encriptado LK1.
config: SessionConfig	Configuración de la sesión.
nEncr: uint	Número de valores SPK (y POPK) adicionales definidos para la encriptación y la posible ulterior desencriptación. El valor máximo es MaxEncr (véase [UIT-T J.1014]).
encrSpk:PubKey[]	Vector con valores SPK adicionales para encriptación.
encrPopk:PubKey[]	Vector con valores POPK adicionales para encriptación.
encrCwUri: ulong	Valor de CWUR a utilizar para encriptación véase la cláusula 8.2.2 de [UIT-T J.1014].

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsStartEncryptSession definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId. El **Anfitrión ECI** deducirá los parámetros importSlotId e importSessionId a partir del valor mh.

NOTA – El mensaje de **Contestación** devuelve el ID de nueva sesión establecida si no se ha producido error alguno.

9.5.2.5.4 Mensaje 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()

- Este mensaje calcula la palabra de control de encriptación.

Definición de los parámetros de la Petición:

sessId: int	Id de la sesión para la que debe calcularse una palabra de control.
cwUri: ulong	El cwUri define las aplicaciones de la palabra de control. Los valores de cwUri se definen en la cláusula 7.5 de [UIT-T J.1014].
nElk: uint	Número de valores Elk en el vector ELK.
elk[24]: SymKey	Vector de valores de claves con encriptación simétrica que se desencriptarán sucesivamente mediante el mecanismo de escalera de claves. El valor elk[nElk-2] es la entrada de datos del campo1 para la autenticación de las propiedades del contenido, tal como se define en la cláusula 8.2.3 de [UIT-T J.1014] utilizando la función definida en la cláusula 8.2.4.6 de [UIT-T J.1014].
XT[32]: uchar	Entrada de datos de reserva para el mecanismo de la palabra de control, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
rkIndx: uint	Identifica si en el cálculo de la palabra de control se utiliza la clave de sesión aleatoria actual (rkIndx==0) o la siguiente (rkIndx==1).
field2: Field2	Contenido relativo a las propiedades del contenido de mayor tamaño no autenticado en el campo1, tal como se define en cláusula 8.2.3 de [UIT-T J.1014].
cwIndx: uint	Índice de la palabra de control a calcular: 0 para la palabra de control par y 1 para la impar; carece de significado para la desencriptación basada en ficheros.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsComputeEncrCw definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.5 Mensaje reqAsAuthEncrSlotConfig

C→H reqAsAuthEncrSlotConfig(uint **sessId**, InputV, uchar **XT**[32], bool **online**, uchar **verifier[16]) →**
H→C resAsAuthEncrSlotConfig()

- Este mensaje autentifica la configuración del intervalo con mecanismos de autenticación (modo encriptación).

Definición de los parámetros de la Petición:

sessId: uint	ID de sesión para la cual debe autenticarse la configuración.
inputV: InputV	Mensaje que contiene la clave pública encriptada del conjunto de chips y el valor r protegido de la firma de la Clave secreta del emisor usado para calcular la AK utilizada a fin de autenticar la configuración del **intervalo AS**.
XT[32]: uchar	Datos de entrada de reserva para el mecanismo de la palabra de control, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
online: booleano	Si es verdadero, se utiliza la clave aleatoria del intervalo para calcular la Clave de autenticación obligando al provisionador a realizar un nuevo cálculo de la Clave de autenticación.
verifier[16]: uchar	Valor con el que reqAsAuthDecrSlotConfig autentifica la configuración del intervalo.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsAuthEncrConfig definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.6 Mensaje reqAsLdUssk

C→H reqAsLdUssk(uint **sessId**, InputV, uchar **XT**[32], bool **online,** uchar **mUssk**[NUSSK]) →
H→C resAsLdUssk()

- Este mensaje carga la clave secreta del **Microservidor** en caso de autenticación asimétrica de los **Clientes ECI** que permitirá decodificar el contenido.

Definición de los parámetros de la Petición:

sessId: uint	ID de sesión para la cual debe cargarse la clave secreta del **Microservidor**.
inputV: InputV	Mensaje que contiene la clave pública encriptada del conjunto de chips y el valor r protegido de la firma de la Clave secreta del emisor usado para calcular la AK utilizada a fin de desencriptar la clave secreta del **Microservidor** que debe cargarse**.**
XT[32]: uchar	Datos de entrada de reserva para el mecanismo de la palabra de control, tal como se define en la cláusula 7.5 de [UIT-T J.1014].
online: booleano	Si es verdadero, se utiliza la clave aleatoria del intervalo para calcular la Clave de autenticación, obligando al provisionador a realizar un nuevo cálculo de la Clave de autenticación.
mUssk[NUSSK]: uchar	Clave secreta encriptada del **Microservidor**.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsLdUssk definida en [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.7 Mensaje reqAsMInikLk1

C→H reqAsMInikLk1(uint **sessId**, ECI_Certificate_Chain **ClCPK) →**
H→C resAsMInikLk1(InputV **inputV)**

- Este mensaje calcula el mensaje de inicialización asimétrica del Microcliente.

Definición de los parámetros de la Petición:

sessId: uint	ID de sesión para la que debe cargarse la clave secreta del **Microservidor**.
ClCPK: ECI_Certificate_Chain	Cadena de certificados objetivo definida en la cláusula 9.5.2.5.2 para cargar la clave pública del conjunto de chips del Microcliente a utilizar para encriptar la clave secreta de sesión entre **Microservidor** y **Microcliente**.

Definición de los parámetros de la Contestación:

inputV: InputV	Clave de sesión MicroDRM encriptada con la Clave pública del conjunto de chips del Microcliente y firmada con la clave secreta del **Microservidor**. Puede ser utilizada por el Microcliente como mensaje de carga de la LK_1 común de la sesión.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsMInikLk1 [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.8 Mensaje reqAsEventCpChange

H→C reqAsEventCpChange(int **sessionId)**

- Este mensaje solicita un cambio de las propiedades del contenido en el contenido importado en una sesión de encriptación.

Definición de los parámetros de la Petición:

sessionId: int	Sesión de encriptación en la que se ha producido un evento de cambio en las propiedades del contenido del contenido importado.

Descripción Semántica:

- Este mensaje es equivalente a la función AS reqAsEventCpChange [UIT-T J.1014]; siendo el **Anfitrión ECI** quien elimina el parámetro slotId.

9.5.2.5.9 Mensaje setAsPermitCPChange

C→H setAsPermitCPChange(int **sessionId**; bool **permit)**

- Este mensaje inicia un cambio en las propiedades del contenido importado en una sesión de encriptación.

Definición de los parámetros de la Petición:

sessionId: int	Sesión de encriptación para permitir una sustitución automática de la palabra de control en un cambio de propiedades del contenido que va a ocurrir o que está pendiente.
permit: booleano	Si es verdadero significa que se ha concedido el permiso, si es falso significa que el permiso no se concede.

Descripción Semántica:

- Esta función es equivalente a la función AS setAsPermitCPChange [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.10 Mensaje setAsSC

C→H setAsSC(int **sessionId**, uint **scramblingControlField**)

- Este mensaje establece el siguiente valor del campo control de aleatorización en una sesión de encriptación.

Definición de los parámetros de la Petición:

sessionId: int	Sesión de encriptación en la que se fija el campo control de aleatorización que se utiliza en el primer punto posible de cambio en el flujo.
scramblingControlField: uint	Valor del campo control de aleatorización; con relación a los valores permitidos y su significado véase la cláusula 9.9 de [UIT-T J.1014].

Descripción Semántica:

- Este mensaje es equivalente a la función AS setAsSC [UIT-T J.1014]; siendo el **Anfitrión ECI** quien proporciona el valor del parámetro slotId.

9.5.2.5.11 Códigos de error de la API de Seguridad avanzada (AS)

Todos los códigos de error para las API de AS se definen en la cláusula 8.2.4.15 de [UIT-T J.1014].

9.5.3 API de Tarjeta inteligente

9.5.3.1 Introducción

La ECI permite que los **Clientes ECI** tengan una interfaz con un único módulo de seguridad local extraíble (**Tarjeta inteligente**). Los **Clientes ECI** pueden crear un canal seguro entre el **Cliente ECI** y la **Tarjeta inteligente** o (desde una perspectiva amplia de seguridad) directamente desde la **Tarjeta inteligente** con el bloque de Seguridad Avanzada para dotar de la máxima robustez a la protección de las palabras de control. La **ECI** no define información detallada sobre los protocolos de los intercambios necesarios para la gestión de claves sino que estos están completamente definidos en el sistema CA/DRM sobre la base de la API del bloque de **Seguridad Avanzada** como se define en [UIT-T J.1014].

Los **CPE** conformes con la **ECI** pueden tener una o varias ranuras para tarjetas. El **Anfitrión ECI** gestiona los lectores de tarjetas de forma completamente transparente para los **Cliente ECI**. El **Anfitrión ECI** establece una correspondencia entre cualquier **Tarjeta inteligente** que se inserte y los **Clientes ECI** disponibles. A tal fin, los **Clientes ECI** publican una lista de especificadores de tarjetas para el **Anfitrión ECI**. El **Anfitrión ECI** gestiona cualquier posible conflicto de acceso entre **Clientes ECI** que deseen acceder a la misma **Tarjeta inteligente**. El **Anfitrión ECI** gestiona además las controversias relacionadas con los lectores de tarjetas.

9.5.3.2 Especificaciones básicas

Esta cláusula proporciona las normas y especificaciones básicas que deberán cumplir el hardware del lector de tarjetas del CPE y los controladores asociados, así como el software del **Anfitrión ECI**.

Las características físicas de un lector de tarjetas de un **CPE** pueden estar basadas en necesidades identificadas por el mercado. El formato predominante de las tarjetas de acceso condicional es ID-1 (tamaño de una tarjeta de crédito), aunque también se usan tarjetas con el formato ID-000 (SIM). Véase como referencia [ISO/CEI 7816-1], [ISO/CEI 7816-2] e [ISO/CEI 14496-12].

Un lector de tarjetas ordinario de **CPE** cumplirá la cláusula 5 de [ISO/CEI 7816-3] que recoge que como mínimo debe soportarse el funcionamiento de clase A (5V) y B (3V). Se deberán soportar las siguientes configuraciones de patillas: C1 (VCC), C2 (RST), C3 (CLK), C5 (GND) y C7 (I/O).

El **Anfitrión ECI** puede soportar lectores de tarjetas que no sean conformes con lo arriba indicado. Esos lectores de tarjetas tendrán una marca clara que los identifique y evite que el **Usuario** los confunda con lectores de tarjeta ordinarios de la **ECI**.

El **Anfitrión ECI** y el hardware del lector de tarjetas del **CPE** soportarán las características pertinentes de la **ECI** definidas en las cláusulas 6 y 12 de [ISO/CEI 7816-2]. El **Anfitrión ECI** inicializará cualquier tarjeta que se inserte utilizando los procedimientos definidos en [ISO/CEI 7816-2].

El **Anfitrión ECI** implementará la funcionalidad [ISO/CEI 7816-3] según sea necesario para cumplir lo especificado en la presente Recomendación. El **Anfitrión ECI** soportará [ISO/CEI 7816-5] en la medida necesaria para cumplir la funcionalidad de obtención del AID (identificador de aplicación) según se define en la cláusula 9.5.3.3 siguiente.

9.5.3.3 Gestión del acceso a una Tarjeta inteligente

Antes de inicializar una conexión con un **Cliente ECI**, el **Anfitrión ECI** inicializará el protocolo y el lector de tarjeta con arreglo a las cláusulas 6 a 11 de [ISO/CEI 7816-3]. Seleccionará los ajustes adecuados del protocolo, los parámetros de temporización de las comunicaciones y la clase operacional de la **Tarjeta inteligente**.

El **Anfitrión ECI** podrá obtener el AID (Identificador de Aplicación definido en la cláusula 8.2.1.2 de [ISO/CEI 7816-4]) tal como se define en la cláusula 8.2.1 de [ISO/CEI 7816-4] y, recuperándolo de la tarjeta de la forma indicada en la cláusula 8.2.2.1 [ISO/CEI 7816-4] , de los bytes históricos o de la cadena de datos inicial. En el caso de **Tarjetas inteligentes** utilizables para varias aplicaciones, el **Anfitrión ECI** podrá obtener la lista de los AID tal como se define en la cláusula 8.2 de [ISO/CEI 7816-4], y específicamente en las cláusulas 8.2.1.1, 8.2.2 y la subcláusula 8.2.2.3.

El **Anfitrión ECI** utilizará la siguiente lista de identificadores de una tarjeta:

1) Si la tarjeta puede utilizarse para varias aplicaciones de conformidad con [ISO/CEI 7816-4] utilizará como lista de identificadores de tarjeta la lista de AID obtenidos de las plantillas de aplicación de EF.DIR's y los AID directamente representados en EF.DIR.
2) Si a diferencia de lo indicado en el punto anterior la tarjeta no admite varias aplicaciones, se utilizará como único identificador de tarjeta el AID recuperado de los "bytes históricos", como se define en la cláusula 8.1.1 o 8.1.2 de [ISO/CEI 7816-4].
3) Si a diferencia de lo indicado en los dos puntos anteriores, no es posible puede obtener ningún AID, se utilizará como Identificador único de Tarjeta la ATR definida en la cláusula 8.2 de [ISO/CEI 7816-4]. Con el fin de establecer la correspondencia, la ATR se define desde T0 a Kt, excluyendo TCK (si está presente).

Sobre la base de la anterior lista de Identificadores de Tarjeta, el CPE hará las correspondencias con **Clientes ECI**.

Los **Clientes ECI** proporcionarán la lista de Especificadores de identificadores de Tarjeta elegibles si están listos para conectarse a una tarjeta. El atributo Tarjeta exclusiva estará presente para cada Especificador de identificador de Tarjeta e indica que el **Anfitrión ECI** señalizará al **Usuario** la existencia de un conflicto de acceso a una **Tarjeta inteligente.** Esto se produce cuando varios **Clientes ECI** solicitan el acceso a una **Tarjeta inteligente** que se corresponda al Especificador del identificador de Tarjeta y dicha **Tarjeta inteligente** se haya insertado o está presente en uno de los lectores de **Tarjetas inteligentes** del **CPE**.

El **Anfitrión ECI** detectará, y cuando sea posible resolverá, cualquier conflicto de acceso entre la identificación de la tarjeta y los correspondientes **Clientes ECI** con arreglo a las normas siguientes:

- Se considera que una **Tarjeta inteligente** se corresponde con un **Cliente ECI** si uno de los identificadores de tarjeta de su lista de identificadores de tarjetas se corresponde con uno de los Especificadores de identificador de Tarjeta del **Cliente ECI**.
- Si una **Tarjeta inteligente** se corresponde con varios **Clientes ECI** y ninguno de los **Clientes ECI** requiere acceso exclusivo, se establece una sesión de tarjeta en el orden siguiente:
  - En primer lugar, se establecerá una sesión de tarjeta para el **Cliente ECI** que haya tenido la sesión más reciente con la tarjeta.
  - Si no existe ningún **Cliente ECI** de ese tipo o no se reconoce que la tarjeta haya sido insertada previamente en el lector de tarjetas del **CPE**, puede establecerse una sesión de tarjeta mediante un algoritmo que selecciona el **Anfitrión ECI**.
- Un **Cliente ECI** podrá desconectar una sesión de **Tarjeta inteligente** si no puede operar con la **Tarjeta inteligente** de forma que el **Anfitrión ECI** pueda establecer una correspondencia entre la misma y otros **Clientes ECI** que puedan intentar utilizarla.

Los **Clientes ECI** podrán gestionar eventos de "conexión" y "desconexión" generados por el **Anfitrión ECI** en una sesión de **Tarjeta inteligente**.

9.5.3.4 Gestión de conflictos de acceso asociados al lector de Tarjetas inteligentes

En esta cláusula se definen funcionalidades para la resolución de conflictos de aplicaciones de los **Anfitriones ECI** para la gestión de disputas de acceso entre clientes y los lectores de tarjetas disponibles para acceder a **Tarjetas inteligentes**.

Cuando se accede a **Tarjetas inteligentes** a través de un lector de tarjetas (sesión de **Tarjeta inteligente**) el **Cliente ECI** establecerá la prioridad de la sesión de **Tarjeta inteligente**. Los posibles valores son los siguientes:

- **Activa**: utilizada para una función primaria que si se interrumpe genera al Usuario una situación indeseable. Un ejemplo de ello es una sesión de visualización solicitada por el Usuario o una sesión de grabación previamente programada por el **Usuario**.
- **De fondo**: se utiliza para un procesamiento en segundo plano que puede interrumpirse si es necesario (es el estado por defecto). Un ejemplo es el procesamiento de mensajes EMM para la adquisición de futuros derechos de acceso.

Un **Cliente ECI** podrá solicitar que se inserte una **Tarjeta inteligente** (lo que implica un uso activo) con referencias a una o más **Asas de medios** o a una cadena que indique que la aplicación que requiere la tarjeta en cuestión no es necesaria para un **Distintivo de medios** específico.

El **Anfitrión ECI** dirigirá al **Usuario,** mediante las orientaciones siguientes, a un lector de tarjetas adecuado si el **Cliente ECI** solicita una tarjeta:

- Intentará dirigirlo a un lector de tarjetas libre, si está disponible.
- En caso de que no se disponga de un lector libre, intentará dirigirlo a un lector de respaldo.

- Si no hay disponibles lectores de respaldo o libres, debería intentar dirigirlo a un lector en modo activo que cause al **Usuario** el menor perjuicio posible, utilizando información de la aplicación/**Cliente ECI** de las sesiones actualmente activas de esos lectores.

El proceso anterior puede implicar que el **Anfitrión ECI** utilice información adicional para establecer la correspondencia entre la tarjeta y el tipo de lector más adecuado (por ejemplo, por sus dimensiones físicas), al asociar al **Cliente ECI** un tipo de lector que se adapte a los requisitos necesarios para una conexión exitosa con el **Cliente ECI** (asumiendo que en el futuro volverá a insertarse el mismo tipo de tarjeta). A tal fin, el **Anfitrión ECI** puede utilizar políticas propias.

9.5.3.5 API de gestión de la sesión de Tarjeta inteligente

9.5.3.5.1 Generalidades

La API de gestión de la sesión de **Tarjeta inteligente** proporcionará a los Clientes acceso gestionado a las **Tarjetas inteligentes** tal como se define en las cláusulas 9.5.3.3 y 9.5.3.4.

Los mensajes de la API disponibles para la gestión de una sesión de **Tarjeta inteligente** figuran en el Cuadro 9.5.3.5.1-1.

Cuadro 9.5.3.5.1-1 – Mensajes de la API de gestión de una sesión de Tarjeta inteligente

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setCardMatch	set	C→H	0x0	Establece la lista de especificadores de identificación de tarjetas para el **Cliente ECI**.
callCardSessionPrio	call	C→H	0x1	Establece la prioridad de la sesión de la **Tarjeta inteligente**.
getCardConnStatus	get	H→C	0x2	Proporciona el estado de la conexión de la tarjeta.
reqCardConOpen	A	H→C	0x3	Informa al **Cliente ECI** que se ha abierto una sesión de tarjeta.
reqCardConClose	A	H→C	0x4	Informa al **Cliente ECI** que se ha cerrado una sesión de tarjeta.
reqCardConClose	A	C→H	0x5	Informe al **Anfitrión ECI** que un **Cliente ECI** desea finalizar una sesión con la tarjeta a la que está conectado.

9.5.3.5.2 Mensaje setCardMatch

C→H setCardMatch(uint **matchListLenth**, CardSpecifier **matchList[])**

- Este mensaje permite al **Cliente ECI** indicar con qué identificadores de tarjetas desea conectarse.

Definición de las propiedades de CardMatch (concordancia de tarjeta)

matchListLength: uint	Longitud de la matchList (lista de concordancias) en términos de especificadores.
matchList: CardSpecifier[]	Véase el Cuadro 9.5.3.6.1-1: Mensajes de comunicación de la **Tarjeta inteligente**. El **Anfitrión ECI** utilizará esta lista para establecer la correspondencia entre las **Tarjetas inteligentes** conectadas y el **Cliente ECI** con arreglo a lo indicado en la cláusula 9.5.3.3. En el Cuadro 9.5.3.5.2-1 se definen los tipos y en el Cuadro 9.5.3.5.2-2 se muestran los valores del campo specifierType (tipo de especificador).

Cuadro 9.5.3.5.2-1 – Definiciones de tipo para especificador de la tarjeta inteligente

```
#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;
```

Cuadro 9.5.3.5.2-2 – Tipo del especificador de la Tarjeta inteligente

Nombre	Valor	Descripción
CardSpecifierATR	0x01	El especificador de tarjeta es de tipo ATR. Una tarjeta se corresponde con un especificador si el campo **atrLen** es idéntico a la longitud ATR de la tarjeta y los bytes ATR de la tarjeta concuerdan con los primeros bytes **atrLen** del campo **atr**. El ATR de una tarjeta se define en la cláusula 9.5.3.5.3, T0..TCK.
CardSpecifierAID	0x02	El especificador de tarjeta es de tipo AID. Una tarjeta se corresponde con un especificador si el campo **aidLen** es idéntico a la longitud AID de la tarjeta y los bytes AID de la tarjeta concuerdan con los primeros bytes **aidLen** del campo **aid**. El AID de una tarjeta se define en la cláusula 9.5.3.3.
RFU	Otros	Reservado para uso futuro.

Precondiciones:

1) El **Cliente ECI** está preparado para responder a los mensajes invCardConOpen y invCardConClose si **matchListLength** > 0.

Condiciones posteriores:

1) El **Anfitrión ECI** comparará cualquier tarjeta que se inserte en un lector de tarjetas con el **Cliente ECI**, tal como se define en 9.5.3.3. En caso de que exista una correspondencia, abrirá una sesión de tarjeta con el **Cliente ECI** como se define en la cláusula 9.5.3.5.5.
2) El **Anfitrión ECI** no desactivará una sesión de tarjeta en curso si en la nueva matchList no ofrece concordancia alguna con la **Tarjeta inteligente** actualmente conectada. El **Cliente ECI** utilizará el mensaje reqCardConnClose para ese fin.

9.5.3.5.3 Mensaje callCardSessionPrio

C→H callCardSessionPrio(uchar **priority**, uint **nrMh**, ushort **mH[]**, char ***clientApplication)**

- Este mensaje actualiza la prioridad de la sesión de tarjeta y proporciona al **Anfitrión ECI** la lista de **Asas de medios** mH y el motivo interno del **Cliente ECI** para solicitar o tener una sesión de tarjeta Activa.

Definición de parámetros de la llamada

priority: uchar	Prioridad de la sesión de tarjeta que requiere el **Cliente ECI**. Los valores se definen en el Cuadro 9.5.3.5.3-1.
nrMh: uint	Número de **Asas de medios** de una sesión Activa con la tarjeta.
mH: ushort	Lista de **Asas de medios** que necesita una sesión Activa con una **Tarjeta inteligente**.
clientApplication: char *	Cadena terminada con un carácter nulo que incluye el motivo por el que el **Cliente ECI** solicita una sesión activa con una **Tarjeta inteligente** no relacionada con una actividad del **Distintivo de medios**. Dicho requisito no existe si el puntero es NULL (nulo). Si el puntero no es NULL el valor de la cadena tendrá significado para el **Usuario**. El número máximo de caracteres que pueden mostrarse es 40.

Cuadro 9.5.3.5.3-1 – Valores de prioridad de la sesión de Tarjeta inteligente

Nombre	Valor	Descripción
CardPriorityBackground	0x01	El requisito de prioridad de la tarjeta del **Cliente ECI** es de fondo y se define en la cláusula 9.5.3.4.
CardPriorityActive	0x02	El requisito de prioridad de la tarjeta del **Cliente ECI** es Activa y se define en la cláusula 9.5.3.4.
RFU	Otros	Reservado para uso futuro.

Postcondiciones:

1) El **Anfitrión ECI** gestionará la sesión de tarjeta como se define en la cláusula 9.5.3.4 con arreglo a la **prioridad** y utilizará **mh** y **clientApplication** para resolver los conflictos de acceso a lectores de tarjeta a través de la interfaz de **Usuario**, si es necesario.

9.5.3.5.4 Mensaje getCardConnStatus

C→H uchar**getCardConStatus()**

- Este mensaje devuelve el estado de conexión de la sesión en curso con una **Tarjeta inteligente**.

Definición de propiedades: véase el Cuadro 9.5.3.5.4-1.

Cuadro 9.5.3.5.4-1 – Valores del estado de conexión de la tarjeta

Nombre	Valor	Descripción
CardConNo	0x00	El **Cliente ECI** no tiene una sesión con una **Tarjeta inteligente**.
CardConYes	0x01	El **Cliente ECI** tiene una sesión con una **Tarjeta inteligente**.
RFU	Otros	Reservado para uso futuro.

9.5.3.5.5 Mensaje reqCCardConOpen

H→C reqCCardConOpen() →
C→H resCardConOpen()

- Este mensaje permite al **Anfitrión ECI** informar al **Cliente ECI** sobre un nuevo evento de conexión de sesión con una tarjeta; el **Cliente ECI** responde confirmando el evento en proceso.

Precondiciones a la Petición:

1) Debe establecerse una sesión con el **Cliente ECI** con arreglo a la cláusula 9.5.3.3.

Postcondiciones a la Contestación:

1) El **Cliente ECI** gestionará la prioridad de la sesión con arreglo a los requisitos establecidos en la cláusula 9.5.3.4.

2) El **Cliente ECI** cerrará la sesión si esta no está justificada para la tarjeta, tal como se define en la cláusula 9.5.3.3.

9.5.3.5.6 Mensaje reqCCardConClose

H→C reqCCardConClose () →
C→H resCardConClose ()

- Este mensaje permite al **Anfitrión ECI** informar al **Cliente ECI** que se ha cerrado la sesión con la tarjeta. El **Cliente ECI** contesta confirmando que el procesamiento del evento.

Precondiciones a la Petición:

1) La tarjeta ha sido extraída del lector o bien se ha producido un funcionamiento defectuoso del subsistema del lector de tarjetas que ha causado la pérdida de la conexión.

Postcondiciones a la Contestación:

1) La **Contestación** del **Cliente ECI** confirma que el **Cliente ECI** ha procesado el evento y está listo para aceptar una nueva conexión de tarjeta tal como define CardMatch (concordancia de tarjeta).

9.5.3.5.7 Mensaje reqHCardConClose

C→H reqHCardConClose() →
H→C reqHCardConClose ()

- Este mensaje permite al **Cliente ECI** indicar al **Anfitrión ECI** que ya no tiene motivo alguno para mantener la interacción con la **Tarjeta inteligente** conectada.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** conecta la **Tarjeta inteligente** con otro **Cliente ECI** concordante tal como se define en la cláusula 9.5.3.3 y no intentará conectar esa tarjeta con el **Cliente ECI** (reinicios y ciclos energéticos pendientes).
2) El **Anfitrión ECI** esperará hasta la recepción de la **Contestación** antes de volver a conectar otra **Tarjeta inteligente** concordante con el **Cliente ECI**.

9.5.3.6 Definición de mensajes de la API de comunicaciones de la Tarjeta inteligente

9.5.3.6.1 Generalidades

La API de **Contestación** e Instrucciones de **Tarjeta inteligente** proporcionará las primitivas de la sesión de comunicación entre un **Cliente ECI** y una **Tarjeta inteligente** en el contexto de una sesión de **Tarjeta inteligente** abierta y gestionada por el **Anfitrión ECI**. El **Cliente ECI** puede realizar intercambios Instrucción/Contestación [ISO/CEI 7816-3] con el **Anfitrión ECI** a nivel de las APDU (véase la Nota), tal como se define en la cláusula 12 de [ISO/CEI 7816-3]. El **Cliente ECI** tiene acceso a todas las funciones de gestión de la **Tarjeta inteligente** y puede reinicializar y restablecer los valores iniciales con parámetros a medida, si ello es necesario, así como obtener los valores de la comunicación. Los mensajes de la API **ECI** se definen en el Cuadro 9.5.3.6.1-1.

NOTA – También permite intercambios contemplados en el protocolo para T=0 a nivel de TPDU mediante el uso de intercambios breves de instrucciones y contestaciones en la interfaz a nivel de ADPU.

Cuadro 9.5.3.6.1-1 – Mensajes de la API de comunicaciones de Tarjeta inteligente

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqCardCmdRes	A	C→H	0x6	Envía instrucciones a la tarjeta, obtiene la contestación de la tarjeta.
reqCardReInit	A	C→H	0x7	Restablece los valores originales de la tarjeta (en frío o en caliente) y devuelve la secuencia de inicialización con los últimos ajustes de preferencia de la inicialización.
callCardSetProp	set	H→C	0x8	Fija los parámetros de comunicación de la tarjeta.
callCardGetProp	get	H→C	0x9	Obtiene los parámetros/propiedades de comunicación de la tarjeta.

9.5.3.6.2 Mensaje reqCardCmdRes

C→H reqCardCmdRes(byte **nodeAddrByte**, uint **cmdApduLen**, byte **cmdApdu[]) →**
H→C resCardCmdRes(uint **resApduLen,** byte **resApdu[])**

- Como se define en la cláusula 12 de [ISO/CEI 7816-3], este mensaje envía una APDU de instrucciones a la **Tarjeta inteligente** a través del **Anfitrión ECI** y obtiene una ADPU de contestación. Los códigos de error conexos se definen en el Cuadro 9.5.3.6.2-1.

Definición de los parámetros de la Petición:

nodeAddrByte: byte	Byte de dirección del nodo para el valor correspondiente al protocolo T=1 del protocolo de **Tarjeta inteligente** establecido, tal como se define en la cláusula 11.3.2.1 de [ISO/CEI 7810]. Este parámetro se ignora si el valor correspondiente al protocolo de la **Tarjeta inteligente** es T=0.
cmdApduLen: unit	Longitud en bytes de la APDU cmd (instrucción). Nótese que la longitud interna de **cmdApdu** no superará el valor de **cmdAduLen**.
cmdAdpu: byte []	APDU de instrucción a enviar a la tarjeta. El **Anfitrión ECI** ignora los bytes en exceso del campo cmdApdu.

Definición de los parámetros de la Contestación:

resApduLen: uint	Longitud en bytes de la ADPU de **Contestación**.
resAdpu: byte []	ADPU de **Contestación** recibida de la tarjeta.

Precondiciones a la Petición:

1) El **Cliente ECI** mantiene abierta una sesión de **Tarjeta inteligente**.
2) La reqCardCmdRes anterior ha dado lugar a una resCardCmdRes o bien la conexión se ha (re)inicializado.

Cuadro 9.5.3.6.2-1 – Códigos de error de resCardCmdRes

Nombre	Descripción
ErrCardConnOpenNot	Véase el Cuadro 9.5.3.7-1.
ErrCardConnFail	

9.5.3.6.3 Mensaje reqCardReInit

C→H reqCardReInit(uchar **resetMode) →**
H→C resCardReInit()

- Este mensaje solicita al **Anfitrión ECI** que reinicie la **Tarjeta inteligente** mediante el resetMode y adopte los valores de ajuste preferentes de la última conexión de tarjeta. La **Contestación** se envía de vuelta una vez completado el proceso (o si se ha producido una falla). Los códigos de error se definen en el Cuadro 9.5.3.6.3-2.

Definición de los parámetros de la Petición:

resetMode: uchar	Véase el Cuadro 9.5.3.6.3-1.

Cuadro 9.5.3.6.3-1 – Valores del resetMode de tarjeta

Nombre	Valor	Descripción
CardResetCold	0x01	Se realiza un reinicio en frío y se reinicializará la tarjeta como si se tratara de su primera activación (véase la cláusula 6.2.3 de [ISO/CEI 7816-1]).
CardResetWarm	0x02	Se realiza un reinicio en caliente, se reinicializarán los parámetros de temporización de las comunicaciones (véase la cláusula 6.2.3 de [ISO/CEI 7816-3]) y se volverá a ejecutar la "el protocolo y la selección de parámetros" tal como se define en la cláusula 9 de [ISO/CEI 7816-3], si procede. Puede utilizarse específicamente para la sustitución de los parámetros de temporización de la interfaz por un valor preferido del **Cliente ECI**.
RFU	Otros	Reservado para uso futuro.

Precondiciones a la Petición:

1) El **Cliente ECI** tiene abierta una sesión de **Tarjeta inteligente**.

Postcondiciones a la Contestación:

1) La **Contestación** indica el establecimiento satisfactorio del protocolo de la interfaz y de los valores de los parámetros.

Cuadro 9.5.3.6.3-2 – Códigos de error de resCardCmdRes

Nombre	Descripción
ErrCardConnOpenNot	Véase el Cuadro 9.5.3.7-1.
ErrCardConnFail	

9.5.3.6.4 Mensaje callCardSetProp

C→H callCardSetProp (ushort **propTag**, uint**valueLen**, byte ***propValue)**

- Este mensaje fija en **propValue** la propiedad grabable indicada mediante la **propTag** de la interfaz de la **Tarjeta inteligente**.

Definición de los parámetros de la Petición:

propTag: ushort	Etiqueta de la propiedad del protocolo de comunicación de tarjeta a sustituir. Los valores se definen en el Cuadro 9.5.3.6.5-2.
valueLen: uint	Longitud en bytes del campo paramValue.
propValue: byte *	Puntero al valor de la propiedad a grabar en el parámetro indicado por propTag.

Cuadro 9.5.3.6.4-1 – Códigos de error de callCardSetProp

Nombre	Descripción
ErrCardConnOpenNot	Véase el Cuadro 9.5.3.7-1.

9.5.3.6.5 Mensaje callCardGetProp

C→H callCardGetPropf(ushort **propTag**, uint **valueLen**, byte ***propValue)**

- Este mensaje lee en **propValue** la propiedad accesible indicada por la **propTag** de la interfaz de la **Tarjeta inteligente**. Los códigos de error conexos se definen en el Cuadro 9.5.3.6.5-1.

Definición de los parámetros de la Petición:

propTag: ushort	Etiqueta de la propiedad del protocolo de comunicación de tarjeta a sustituir. Los valores se definen en el Cuadro 9.5.3.6.5-2.
valueLen: uint	Longitud máxima en bytes del campo propValue. Los bytes en exceso relativos a la propiedad no se copian en propValue.
propValue: byte *	Puntero al valor de propiedad solicitada.

Cuadro 9.5.3.6.5-1 – Códigos de error de callCardSetProp

Nombre	Descripción
ErrCardConnOpenNot	Véase el Cuadro 9.5.3.7-1.

Cuadro 9.5.3.6.5-2 – Valores y semántica de las etiquetas de la API de tarjetas para las propiedades del protocolo de tarjeta

Nombre	Valor de la etiqueta	Descripción
CardPropClass	0x0001	Un byte. Valor de Clase A=0x01, Clase B=0x02, Clase=0x03. Los restantes valores quedan reservados para uso futuro. Sólo lectura.
CardPropAtrLen	0x0002	Un byte. Longitud en bytes del ATR de la tarjeta en **CardPropAtr**. Solo lectura.
CardPropAtr	0x0003	Cadena de bytes, máximo 16 bytes. Reinicio en frío del ATR de la tarjeta. Solo lectura.
CardPropPpsExch	0x0004	La tarjeta y la interfaz han completado con éxito un intercambio PPS si no es igual a 0x00. Solo lectura.
CardPropPpsVal	0x0004	Un byte. Valor del resultado del intercambio PPS de tarjeta de PPS1. La presente Recomendación no admite otros valores. Solo lectura.
CardPropTAEff	0x0005	Un byte. Valor efectivo de TA aplicado a la temporización de reloj en la interfaz. Solo lectura.
CardPropTCEff	0x0006	Un byte. Valor efectivo de TC aplicado a la temporización de reloj en la interfaz. Solo lectura.
CardPropProt	0x0007	Un byte. Indica el protocolo seleccionado por el dispositivo de la interfaz para la comunicación con la tarjeta. Los valores se definen en la cláusula 8.2.3 de[ISO/CEI 7816-3], campo "T". El valor 0x00 indica el protocolo para T=0, el valor 0x01 indica el protocolo para T=1. Pueden darse otros valores (hasta 0x0E). Solo lectura.
CardPropT1IFSC	0x0008	Un byte. El valor de IFCS (tamaño del campo de información de la tarjeta) en el protocolo codificado correspondiente a T=1 se define en la cláusula 11.4.2 de [ISO/CEI 7816-3]. Solo lectura.
CardPropT1IFSD	0x0009	Un byte. El valor de IFSD (Tamaño del campo de información del dispositivo = lector de tarjetas) en el protocolo codificado correspondiente a T=1 se define en la cláusula 11.4.2 de [ISO/CEI 7816-3]. Solo lectura.
CardPropAidListLen	0x000A	Un byte: longitud de la lista de los AID de tarjeta obtenida de la tarjeta durante la inicialización. Solo lectura.
CardPropAidList	0x000B	*(byte[MaxAid]): lista de los de la tarjeta obtenidos de la tarjeta durante la inicialización. Solo lectura.
CardPropClassPref	0x0011	Tres bytes. Secuencia de valores preferidos de Clase. Se intentarán establecer en orden los valores de preferencia (sin incumplir safe=ty). Los valores de los 3 bytes están incluidos en **CardPropClass,** donde el valor 0x00 significa "no más preferencias". Lectura y escritura.
CardPropImplClock	0x0012	Valor de **TA** de un byte se aplicará si el bit 5 de $\mathbf{TA}_2$ del ATR indica valores implícitos para la frecuencia de reloj. Lectura y escritura.

Cuadro 9.5.3.6.5-2 – Valores y semántica de las etiquetas de la API de tarjetas para las propiedades del protocolo de tarjeta

Nombre	Valor de la etiqueta	Descripción
CardPropPps1SegLen	0x0013	Un byte. El valor representa un número sin signo binario. El valor mínimo es 0 y el máximo 0x08. Representa el número de valores PPS1 posibles en una negociación de intercambio PPS en **CardPropPps1Seq** tal como se define en la cláusula 9 de [ISO/CEI 7816-3]. Véase la Nota.
CardPropPps1Seq	0x0014	Secuencia de elementos de un byte de longitud máxima 8 que comienza con el valor más deseable para PPS1 a fin de intentar establecerlo en un intercambio PPS. Los valores se definen en la cláusula 9.2 de [ISO/CEI 7816-3]. Lectura y escritura.
CardPropInfdPref	0x0015	Un byte. El valor indica el valor preferido de IFSD a establecer por el dispositivo de la interfaz para el protocolo T1. Lectura y escritura.
RFU	Otros	Reservado para uso futuro.
NOTA – Esta API no soporta los valores para PPS2 y PPS3, no siendo necesario que el **Anfitrión ECI** los soporte. Lectura y escritura.		

9.5.3.7 Códigos de error de la API de Tarjeta inteligente

Los valores de los errores específicos de la API que pueden devolver los mensajes **Contestación** para esta API figuran en el Cuadro 9.5.3.7-1.

Cuadro 9.5.3.7-1 – Códigos de error de la API de Tarjeta inteligente

Nombre	Valor	Descripción
ErrCardOpenNot	-256	No hay establecida ninguna sesión de tarjeta.
ErrCardConnFail	-257	Se ha establecido una sesión de tarjeta pero sin conexión (después del reinicio).
RFU	**Otros**	Reservado para uso futuro.

9.5.4 API de Adquisición del carrusel de datos

9.5.4.1 Generalidades

La API de adquisición del carrusel de datos permite a un **Cliente ECI** obtener información de un carrusel de difusión con formato **ECI** tal como se define en la cláusula 7.7.2. Un **Cliente ECI** puede utilizarla para, entre otras cosas, obtener información de exportación/importación que posiblemente haya sido actualizada.

NOTA – Los carruseles de datos están diseñados para transportar datos cuasi estáticos y no un protocolo de transporte de preferencia para datos provisionales.

Un **Cliente ECI** puede leer directamente datos del carrusel o solicitar al **Anfitrión ECI** que supervise las actualizaciones de un módulo o grupo de elementos del carrusel en el que esté interesado. A efectos de la supervisión ello puede hacerse durante el estado de energía PwrOn (encendido) o en algún periodo especificado en estado de reposo. Se alienta (por motivos de gestión de consumo de energía) que esos periodos coincidan con los periodos de supervisión del **Anfitrión ECI**.

El **Anfitrión ECI** tratará de adquirir los datos solicitados y almacenarlos en un fichero al que el **Cliente ECI** acceda posteriormente a través del sistema de ficheros. El **Anfitrión ECI** proporciona un número mínimo de canales de adquisición paralelos por **Cliente ECI** tal como se define en [b-UIT-T J Supl. 7].

Los mensajes de la API de adquisición del carrusel de datos figuran en el Cuadro 9.5.4.1-1.

Cuadro 9.5.4.1-1 – Mensajes de la API de adquisición del carrusel de datos ECI

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqDCAcqGroupInfo	A	C→H	0x0	El **Cliente ECI** solicita al **Anfitrión ECI** que lea la estructura GroupInfoIndication en el mensaje DSI del carrusel de datos ECI especificado.
reqDCAcqModule	A	C→H	0x1	El **Cliente ECI** solicita al **Anfitrión ECI** que adquiera un módulo del carrusel de datos **ECI** específico y lo incluya en un fichero utilizando parámetros de un filtro de módulo y varios modos.

9.5.4.2 Mensaje reqDCAcqGroupInfo

C→H reqDCAcqGroupInfo (uint **operatorId**, uint **platformId) →**
H→C resDCAcqGroupInfo (byte **gii[])**

- El **Cliente ECI** solicita al **Anfitrión ECI** que lea la estructura GroupInfoIndicationclen en el mensaje DSI del carrusel de datos **ECI** especificado. Los códigos de error conexos se definen en el Cuadro 9.5.4.2-1.

Definición de los parámetros de la Petición:

operatorId: uint	ID del **Operador** de 20 bits incluido en la estructura ECI_carousel_id transportada en el data_broadcast_id_descriptor() del PSI (véase la cláusula 7.7.2.4).
platformId: uint	ID de la **Operación de Plataforma** de 20 bits incluido en la estructura ECI_carousel_id transportada en el data_broadcast_id_descriptor() del PSI (véase la cláusula 7.7.2.4).

Definición de los parámetros de la Contestación:

gii: byte[]	Matriz de bytes que transporta la estructura GoupInfoIndication como figura en el DSI del carrusel, tal como se define para DVB DSM-CC [ETSI EN 301 192].

Información de la Semántica:

- El **Anfitrión ECI** solo proporciona acceso a carruseles de clientes que han sido cargados.

Cuadro 9.5.4.2-1 – Códigos de error de reqDCGroupInfo

Nombre	Descripción
ErrDCAcqNetwAccessResource	Véase el Cuadro 9.5.4.4-1.
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	

9.5.4.3 Mensaje 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()

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** la adquisición de un módulo específico de carrusel de datos ECI en un fichero utilizando parámetros de un filtro de módulo y varios modos.

Definición de los parámetros de la Petición:

aid: uchar	Número del filtro de adquisición. Un **Cliente ECI** puede tener un máximo de tres filtros de adquisición activos (valores 0 .. 2).
fname: fileName	Nombre del fichero en el que se copiarán los datos del módulo carrusel a adquirir. Se sobrescribe sobre datos preexistentes.
oId: uint	ID del **Operador** de 20 bits, tal como figura en la estructura de ECI_carousel_id transportada en el data_broadcast_id_descriptor() del PSI (véase la cláusula 7.7.2.4).
pId: uint	ID de la **Operación de Plataforma** de 20 bits, tal como figura en la estructura ECI_carousel_id transportada en el data_broadcast_id_descriptor() del PSI (véase la cláusula 7.7.2.4).
dType: byte	Este campo debería concordar con el campo tipo de Descriptor del grupo de módulos definido en el Cuadro 7.7.2-3.
model: uint	Transporta un valor sin signo de 16 bits que debería concordar con el campo modelo del compatibilityDescriptor del grupo a adquirir. Véase el Cuadro 7.7.2.4-1.
version: uint	Transporta un valor sin signo de 16 bits que debería concordar (filtro positivo) o no (filtro negativo) o ser descartado por motivos de concordancia, con el campo versión que figura en el compatibilityDescriptor del grupo a adquirir, en función de los bits 0 y 1 del parámetro **modo**. Véase el Cuadro 7.7.2.4-1.
index: uint	Índice del módulo al que acceder en el grupo. Este parámetro será interpretado con arreglo al bit 1 del parámetro **modo**.
mode: uint	Parámetro compuesto de varios campos: **bit 0**: señaliza el filtrado positivo o negativo de la **versión**: 0b0 significa filtrado positivo y 0b1 filtrado negativo; **bit 1**: señaliza si el filtrado de la versión debe ignorarse (valor 0b1) o no (0b0); **bit 2**: señaliza si el índice debe ignorarse (valor 1) y debe adquirirse cualquier módulo (para carruseles de un solo módulo) o si es necesario utilizar el índice (módulo numberOfModules, véase el Cuadro 7.7.2.6-1); **bit 29**: si se pone a 1, el **Anfitrión ECI** realizará la adquisición durante el periodo de reposo verificando el carrusel con arreglo a sus propios requisitos de adquisición, debiendo continuar la adquisición hasta una ulterior notificación en los modos reposo (standby) y encendido (powerOn) una vez hayan sido adquirido los datos solicitados; **bit 30**: señaliza si la adquisición asumirá que el carrusel de datos (datacarousel) está en funcionamiento y la adquisición debe completarse durante el periodo de actividad normal del carrusel (valor 0b0) o si la adquisición deberá realizarse cómo y cuándo sea posible y cuándo exista correspondencia alguna con el filtro de adquisición (0b1) (es decir, esperar hasta que los datos se presenten por sí mismos); **bit 31:** habilita (valor 0b1) o inhabilita (valor 0b0) la adquisición con este filtro **aid**.

Precondiciones a la Contestación:

1) El módulo carrusel solicitado ha sido adquirido, se ha encontrado un error en el sistema de ficheros, o bien, si el bit 30 **mode** está puesto a 1, se ha detectado un problema relativo a la adquisición.

2) El **Anfitrión ECI** está en el estado PwerOn (encendido). Es decir, no se despierta al **Cliente ECI** en relación con una adquisición durante el periodo de reposo.

Postcondiciones a la Contestación:

1) El fichero contiene el módulo especificado o bien, se ha producido un error.

2) Cuando el bit 30 del parámetro **mode** está puesto a 1, no pueden producirse errores de adquisición.

Información de la Semántica:

- El **Anfitrión ECI** solo proporciona acceso a carruseles de **Clientes ECI** que han sido cargados y para los cuales realiza la supervisión de datos de difusión para los fines del **Anfitrión ECI**.
- Si no está puesto a 1, no se realizará adquisición alguna en estado de reposo. Los **Clientes ECI** que deseen crear su propio plan de adquisiciones pueden hacerlo mediante la API para despertar del estado de reposo (Wakeup) descrita en la cláusula 9.4.7.3.

- El **Anfitrión ECI** proporcionará una **Contestación** "trivial" en caso de petición con el bit 31 mode puesto a cero.

Los códigos de error conexos figuran en el Cuadro 9.5.4.3-1.

Cuadro 9.5.4.3-1 – Códigos de error de reqDCAcqModule

Nombre	Descripción
ErrDCAcqNetwAccessResource	Véase el Cuadro 9.5.4.4-1.
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	
ErrDCAcqCarNoGroup	
ErrDCAcqCarNoModule	
ErrDCAcqCarTimeout	
ErrDCAcqFileSystemFailure	
ErrDCAcqFileQuotaExceeded	

9.5.4.4 Códigos de error para la API de adquisición del carrusel de datos

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes de **Contestación** de esta API figuran en el Cuadro 9.5.4.4-1.

Cuadro 9.5.4.4-1 – Códigos de error de la API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrDCAcqNetwAccessResource	-256	Véase el Cuadro 9.6.2.3.7-1.
ErrDCAcqNetwAccessFail	-257	Véase el Cuadro 9.6.2.3.7-1.
ErrDCAcqNoCarousel	-258	En las redes de difusión accesibles por el **Anfitrión ECI** no se ha encontrado ningún carrusel con ID de **Operador** y de **Operación de Plataforma** concordantes.
ErrDCAcqCarNoGroup	-260	Se ha encontrado la estructura groupInfoIndication en el DSI del carrusel, pero no se ha encontrado un grupo concordante.
ErrDCAcqCarNoModule	-261	Se ha detectado el grupo del carrusel (DII) pero no pudo encontrarse un módulo concordante.
ErrDCAcqCarTimeout	-262	Ha vencido un temporizador durante el acceso al DSI, DII o DDB del carrusel.
ErrDCAcqFileSystemFailure	-263	Véase el Cuadro 9.4.5.5-1.
ErrDCAcqFileQuotaExceeded	-264	Véase el Cuadro 9.4.5.5-1.

9.6 Conjunto de las API de acceso a recursos de desencriptación del Anfitrión ECI

9.6.1 API de desencriptación del Anfitrión ECI

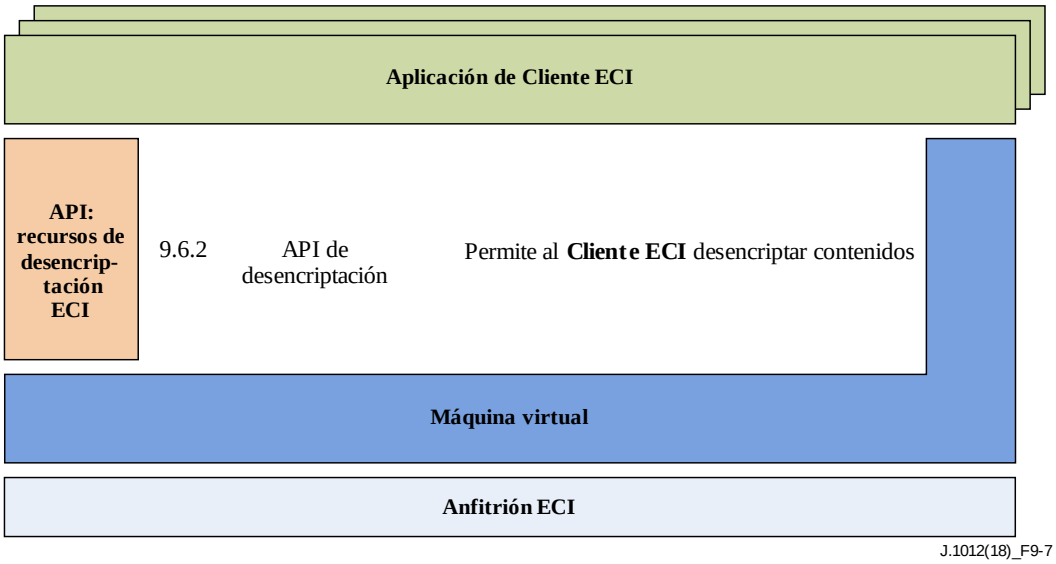


Figura 9.6.1-1 – Representación esquemática de las API definidas en la cláusula 9.6

En el Cuadro 9.6.1-1 figuran las API incluidas en la cláusula 9.6 y la Figura 9.7.1 ilustra la ubicación de las API definidas en la cláusula 9.6 en la arquitectura **ECI**.

Cuadro 9.6.1-1 – Lista de las API definidas en la cláusula 9.6

Cláusula	Nombre de la API	Descripción
9.6.2	API de desencriptación del **Anfitrión ECI**	Permite al **Cliente ECI** entregar al **Anfitrión ECI** información de la URI estándar relacionada con un determinado elemento de contenido.

9.6.2 Definición de la API de desencriptación del Anfitrión ECI

9.6.2.1 Introducción

Las API de desencriptación permiten al **Anfitrión ECI** (por ejemplo, a petición de aplicaciones residentes o descargadas) seleccionar un **Cliente ECI** que concuerde en términos de requisitos de desencriptación del contenido y solicitar que sea desencriptado. Todos los mensajes de desencriptación entre un **Cliente ECI** y un **Anfitrión ECI** se intercambian en el contexto de un **Asa de Medios** que representa el contenido, cualquier red de distribución asociada y los recursos necesarios para su decodificación.

Las API siguientes constituyen el conjunto de las API de desencriptación:

1) API de sesión de medios genéricos para todo tipo de medios, incluidas las funciones de concordancia entre contenido y **Cliente ECI**.
2) Las API de desencriptación del flujo de transporte.
3) Las API de desencriptación de ficheros y de flujos.

9.6.2.2 API de sesión de medios

9.6.2.2.1 Generalidades

El **Cliente ECI** puede anunciar la lista de Especificadores de concordancia en virtud de la cual el **Anfitrión ECI** puede establecer la correspondencia entre aquel y el contenido.

El **Anfitrión ECI** puede solicitar a un **Cliente ECI** concordante que abra una sesión de desaleatorización para un **Asa de Medios**. La apertura de una sesión no implica que comience un proceso de decodificación. Simplemente garantiza la disponibilidad en el **Anfitrión ECI** y en el **Cliente ECI** de recursos necesarios para acceder al contenido y/o los metadatos asociados y llevar a cabo una sesión de desaleatorización. Los **Clientes ECI** deben garantizar el acceso a **Tarjetas inteligentes** u otros recursos necesarios para la desaleatorización del contenido antes de confirmar una sesión. En el Cuadro 9.6.2.2.1-1 figuran las funciones de la API.

Cuadro 9.6.2.2.1-1 – Mensajes de la API de sesión de desencriptación de un Distintivo de medios

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setDcrMhMatch	Set	C→H	0x0	Señaliza al **Anfitrión ECI** los Id (identificadores) que permiten reconocer al **Cliente ECI** para la desencriptación del contenido.
reqDcrMhOpen	A	H→C	0x1	El **Anfitrión ECI** solicita al **Cliente ECI** que abra una sesión de medios de un tipo especificado utilizando un **Asa de Medios**.
reqDcrMhClose	A	H→C	0x2	El **Anfitrión ECI** cierra una sesión de medios con un **Cliente ECI**.
reqDcrMhBcAlloc	A	C→H	0x3	El **Cliente ECI** solicita una sesión de **Asa de Medios** con fines de acceso a una red de difusión.
reqDcrMhCancel	A	C→H	0x4	El **Cliente ECI** cancela una sesión de medios con el **Anfitrión ECI**.

9.6.2.2.2 Mensaje setDcrMhMatch de la API

C→H setDcrMhMatch(uint **matchListLength**, MatchSpecifier **matchList[])**

- Este mensaje permite que el **Cliente ECI** indique al **Anfitrión ECI** cuáles son los Id del sistema de desencriptación para los que puede proporcionar servicios de desencriptación de flujos de transporte.

NOTA – La capacidad real de desencriptación de contenidos puede depender de la suscripción, la situación de pago u otras condiciones.

Definición de la propiedad de SetDcrMhMatch

matchListLength: uint	Longitud de matchList en términos de especificadores.
matchList: MatchSpecifier[].	Cuadro 9.6.2.2.2-1. El **Anfitrión ECI** utilizará esta lista para comparar el contenido y las potenciales capacidades de desencriptación del **Cliente ECI** de conformidad con la cláusula 9.5.3.3. El tipo MatchSpecifier define los especificadores de concordancia. Para que se produzca una correspondencia, todos los campos del MatchSpecifier deben concordar con el contenido.

Cuadro 9.6.2.2.2-1 – Definición de tipos de MatchSpecifier

```
#define MaxMhSubFormat 16;
typedef struct MatchSpecifier {
   uchar decryptIdType; /*véase Cuadro 9.6.2.2.2-2 */
   union decryptId {
      bool     ECI Client ID;
      ushort   dvbCaId;
      byte     uuid[16];
   }
   byte  mhType;
   byte  subFormat[MaxMhSubFormat];
} MatchSpecifier;
```

Cuadro 9.6.2.2.2-2 – Definición del decryptIdType de setDcrMhMatch

Nombre	Valor	Descripción
None	0x00	No concuerda con ningún contenido relacionado con una **Petición** realizada; en caso de apertura de una sesión. indica "sin concordancia".
ClientEciId	0x01	La identificación del **Cliente ECI** puede hacerse sobre la base del Id del **Cliente ECI,** compuesto por los valores de 20 bits (sin incluir los campos tipo y versión) siguientes: <<operator_id,platform_operation_id>,<vendor_id,client_id>> tal como se especifica en la cláusula 7 de la presente Recomendación.
ClientDvbCaId	0x02	El decryptId es un Identificador del sistema de acceso condicional definido en [CEN EN 50221] y [ETSI EN 301 192]. Este valor indica que dvbCaId es la variante utilizada de la unión specifierType. Los valores 'actual+' de dvbCaId se definen en [CEN EN 50221].
ClientUUID	0x03	El decrytId es un ID de DRM tal como define CENC/Dash, que se especifica como un identificador único universal (UUID) [IETF RFC 4122].
RFU	Otros	Reservado para uso futuro.

mhType: unit	Tipo de **Asa de Medios** (modo de desencriptación principal) soportado por el **Cliente ECI** para este ClientEciID.
subFormat: byte[]	Este parámetro permite definir una especificación de un tipo adicional para el **Cliente ECI**. La interpretación de estos bytes es función del **mhType** y se define en el Cuadro 9.6.2.2.2-3.

Cuadro 9.6.2.2.2-3 – Definición del tipo subFormat

Valor de mhType	Semántica del campo subFormat
ISOBMFF	El campo subFormat no contiene ninguna o contiene varias definiciones 4CC secuenciales de valores de marca de caja (box brand) ftyp o styp ISO BMFF adecuados para la decodificación por el **Cliente ECI**. Uno (o más) de esos valores 4CC se compararán a efectos de concordancia con los valores `major_brand` o `compatible_brands[]` de la caja ftyp o styp del contenedor ISO BMFF. El valor 0x0000 de subFormat significa que no existen valores (no existen concordancias), y el valor 0xFFFF de la primera entrada significa que se produce para cualquier valor de marca ("brand") (con independencia de los bytes que sigan).
Otros	Reservado para uso futuro.

Información de la Semántica:

Cuando se intenta representar contenido basado en un flujo de transporte, el **Anfitrión ECI** tratará de identificar correspondencias existentes entre contenido y **Clientes ECI** disponibles utilizando las siguientes normas de ordenación de prioridades:

1) El **Anfitrión ECI** tratará de establecer un conjunto de especificadores de concordancia aplicables utilizando los ID de **Clientes ECI** para ese contenido tal como se define en la cláusula 7.2.2. Si algún ID de **Cliente ECI** aplicable y las propiedades de concordancia asociadas se corresponden con el **MatchSpecifier** (especificador de concordancia) de un **Cliente ECI**, ofrecerá a ese **Cliente ECI** contenido para su desencriptación. Si esa correspondencia existe para varios **Clientes ECI**, el **Anfitrión ECI** utilizará el procedimiento siguiente:

   a) El **Anfitrión ECI** ofrecerá el contenido para su desencriptación al **Cliente ECI** que haya distribuido más recientemente de forma satisfactoria las palabras de control (CW) para la desencriptación de contenido de la misma "fuente de contenido".

   b) Si el primer **Cliente ECI** fracasa en la desencriptación del contenido, intentará utilizar **Clientes ECI** alternativos concordantes ordenados según su historial de desencriptación exitosa más reciente relacionada con la "fuente de contenido".

2) Si el **Anfitrión ECI** no puede identificar ningún ID de **Cliente ECI** para el contenido en cuestión o si ninguno de los **Clientes ECI** según al párrafo anterior puede decodificar el contenido, el **Anfitrión ECI** intentará establecer un conjunto de otros ID para el contenido tal como se define en la cláusula 9.5.4.39.5.4.3. Si la correspondencia solo existe para un **Cliente ECI** y un identificador y las propiedades de concordancia asociadas, el **Anfitrión ECI** ofrecerá a ese **Cliente ECI** el contenido para su desencriptación. Si la correspondencia existe para varios **Clientes ECI**, el **Anfitrión ECI** utilizará el procedimiento siguiente:

   a) El **Anfitrión ECI** ofrecerá el contenido para su desencriptación al **Cliente ECI** que haya desencriptado más recientemente de forma satisfactoria contenido de la misma "fuente de contenido".

   b) Si el primer **Cliente ECI** fracasa en la desencriptación del contenido, intentará utilizar **Clientes ECI** alternativos concordantes ordenados según su historial de desencriptación exitosa más reciente relacionada con la "fuente de contenido".

El término "fuente de contenido" arriba mencionado incluirá como mínimo:

3) Una red o paquete de programas de difusión DVB que genera el flujo de transporte (TS).

4) Un sitio de internet en el que se utilice un navegador que ofrezca referencias al contenido.

9.6.2.2.3 Mensaje reqDcrMhOpen

H→C reqDcrMhOpen(ushort **mH**, MatchSpecifier **match) →**
C→H resDcrMhOpen(ushort **mH)**

- Este mensaje permite al **Anfitrión ECI** solicitar una sesión de desencriptación con el **Cliente ECI**. El **Cliente ECI** debería reservar todos los recursos normalmente necesarios para la desencriptación identificados mediante **mh** y **match**. Los códigos de error conexos se definen en el Cuadro 9.6.2.2.3-1.

Definición de los parámetros de la petición:

mH: ushort	**Asa de Medios** del contenido a desencriptar.
match: MatchSpecifier	Copia del especificador de concordancia (también contiene el tipo de Asas de medios de la sesión).

Definición de los parámetros de la contestación:

mH: ushort	**Asa de Medios** del contenido a desencriptar.

Precondiciones a la Petición:

- El **Anfitrión ECI** ha reservado todos los recursos necesarios para desencriptar el contenido. Para el contenido de un TS ello incluye la sintonización u otros recursos de acceso a la red y el control aplicable, los recursos de demultiplexación y los recursos de desaleatorización para la aplicación de como mínimo una pareja de palabras de control (cw).

Postcondiciones a la Contestación:

- Si el resultado ha sido satisfactorio, el **Cliente ECI** ha reservado para la sesión solicitada todos los recursos normalmente necesarios para la decodificación de contenido. Ello incluye el acceso a cualquier recurso externo (servidores DRM, **tarjetas inteligentes**, etc.) normalmente requeridos para una operación de desencriptación.

NOTA – Se excluyen los recursos necesarios con carácter excepcional o aquellos recursos que normalmente pueden conseguirse cuando se solicitan.

- Si se devuelve ErrDcrUserDelay, el **Cliente ECI** queda a la espera de recibir una entrada del **Usuario** para abrir la sesión (por ejemplo, para obtener acceso a una **Tarjeta inteligente**). El **Anfitrión ECI** debe repetir el envío de la petición reqDcrMhOpen (con los mismos parámetros) hasta que se devuelva un resultado positivo o un error definitivo, o bien alternativamente, pueda enviarse un reqDcrMhClose para terminar la sesión pendiente. El **Cliente ECI** puede realizar la cancelación con reqDcrMhCancel si no consigue la entrada de datos de **Usuario** requerida.

Cuadro 9.6.2.2.3-1 – Códigos de error de reqDcrMhOpen

Nombre	Descripción
ErrDcrUserDelay	Véase el Cuadro 9.6.2.2.7-1.
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.2.4 Mensaje reqDcrMhClose

H→C reqDcrMhClose(ushort **mH) →**
C→H resDcrMhClose(ushort **mH)**

- Este mensaje permite al **Anfitrión ECI** cerrar una sesión de desencriptación con el **Cliente ECI**. El **Cliente ECI** puede liberar entonces los recursos para la sesión.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** para la sesión a cerrar.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** para la sesión cerrada.

Postcondiciones a la Petición:

- El **Cliente ECI** libera los recursos que necesitó específicamente para la sesión.

Postcondiciones a la Contestación:

- El **Anfitrión ECI** puede liberar los recursos relacionados con el **Asa de Medios**.

9.6.2.2.5 Mensaje reqDcrMhBcAlloc

C→H reqDcrMhBcAlloc(byte **networkType[2]**, uchar **priority**, char **reason[80]) →**
H→C resDcrMhBcAlloc(ushort **mH)**

- Este mensaje permite al **Cliente ECI** solicitar la conexión a una red de difusión para la adquisición de datos de seguridad.

Definición de los parámetros de la Petición:

networkType: byte[2]	Tipo de red de difusión a la que accede un **Cliente ECI**; los valores son conformes con el Cuadro 9.6.2.3.6.2-3.
priority: uchar	Prioridad del acceso a la red según se define en el Cuadro 9.6.2.2.5-1.
reason: char[80]	Cadena terminada con nulos de 80 caracteres como máximo que puede presentarse al **Usuario** para resolver cualquier conflicto de recursos en el **Anfitrión ECI** para la resolución de esta petición.

Cuadro 9.6.2.2.5-1 – Definición de la prioridad de acceso a la red de difusión

Nombre	Valor	Descripción
DcrAllocPrioBackground	0x01	Es necesario el acceso para un procesamiento en segundo plano que puede no estar asegurado o ser interrumpido si una tarea de mayor prioridad necesita acceder a los recursos. Por ejemplo, el acceso a EMM o a los datos de renovación de la seguridad en un multiplexador principal.
DcrAllocPrioActivec	0x02	Es necesario el acceso a una función de desaleatorización primaria, que si no se garantiza (o si se interrumpe) genera inconvenientes al **Usuario**. Un ejemplo es una sesión de visualización solicitada por un **Usuario** o una sesión de grabación previamente programada por el **Usuario**.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** para la sesión abierta.

Información de la Semántica:

- El **Anfitrión ECI** puede cancelar la sesión si otra tarea necesita los recursos de acceso a la red con mayor prioridad, en cuyo caso utiliza el mensaje reqDcrMhClose.
- El **Cliente ECI** cerrará la sesión mediante el mensaje reqDcrMhCancel si ya no necesita acceder a la red.

Postcondiciones a la Petición:

1) El **Anfitrión ECI** ha asignado todos los recursos para acceder al tipo de red solicitado.

Postcondiciones a la Contestación:

1) El **Cliente ECI** realizará la sintonización precisa para adquirir un flujo de transporte utilizando el mensaje reqDcrTsRelocate antes de comenzar la adquisición de la sección.

Cuadro 9.6.2.2.5-2 – Códigos de error de reqDcrMhBcAlloc

Nombre	Descripción
ErrDcrNetworkAccessCapability	Véase el Cuadro 9.6.2.2.7-1.
ErrDcrNetworkAccessResource	
ErrDcrPrioOverride	
ErrDcrResourceMissing	

9.6.2.2.6 Mensaje reqDcrMhCancel

C→H reqDcrMhCancel(ushort **mH**, uchar **reason) →**
H→C resDcrMhCancel(ushort **mH)**

- Este mensaje permite al **Cliente ECI** cerrar una sesión de desencriptación con el **Anfitrión ECI**. El **Cliente ECI** ha liberado todos los recursos específicamente necesarios para la sesión.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** para la sesión a cerrar.
reason: uchar	Motivo de cancelación de la sesión de desencriptación. Los valores se definen en el Cuadro 9.6.2.2.6-1.

Cuadro 9.6.2.2.6-1 – Valores de los motivos de reqDcrMhCancel

Nombre	Valor	Descripción
DrcMhUndefined	0x00	Se ha producido un error no definido en el **Cliente ECI** que exige cancelar la sesión.
DcrMhCardMissing	0x01	Es necesario disponer de una **Tarjeta inteligente** para la decodificación pero no pudo (re)conectarse con éxito y ayudar a desencriptar el contenido en un tiempo razonable.
DcrMhServiceMissing	0x02	No se dispone en un tiempo razonable de un servicio (externo al **CPE**) que apoye al **Cliente ECI** en los servicios de desencriptación necesarios para mantener una sesión de desencriptación.
DcrMhResourceMissing	0x03	El **Cliente ECI** no dispone en un tiempo razonable de un recurso (interno al CPE) para la prestación de servicios de desencriptación (sin incluir **DcrMhMmiMissing**).
DcrMhMmiMissing	0x04	El **Cliente ECI** no ha conseguido disponer en un tiempo razonable de un recurso de sesión MMI para la interacción del **Usuario** necesario al objeto de mantener la sesión de desencriptación.
DcrMhAllocTerminate	0x05	El **Asa de Medios** fue asignada en nombre del **Cliente ECI** mediante reqDcrMhBcAlloc pero el **Cliente ECI** ya no la necesita.
RFU	Otros	Reservado para uso futuro.

En [b-UIT-T J Supl. 7] se propone el tiempo razonable para que el **Anfitrión ECI** cancele un **Asa de Medios**.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** para la sesión cancelada.

Precondiciones a la Petición:

- El **Cliente ECI** ha liberado todos los recursos que necesitaba específicamente para la sesión.

Postcondiciones a la Petición:

- El **Anfitrión ECI** puede liberar cualquier recurso relacionado con el **Asa de Medios**.

Postcondiciones a la Contestación:

- El Anfitrión ECI cierra la sesión del Asa de Medios.

9.6.2.2.7 Códigos de error de la API de sesión de medios

Los valores de los errores específicos de la API que pueden devolver los mensajes **Contestación** de esta API, figuran en el Cuadro 9.6.2.2.7-1.

Cuadro 9.6.2.2.7-1 – Códigos de error de la API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrDcrUserDelay	-256	Se ha producido una demora prolongada en espera de una entrada procedente del **Usuario** para completar la operación. La operación no se ha completado.
ErrDcrCardMissing	-257	La **Tarjeta inteligente** necesaria para la sesión no es accesible o no está disponible.
ErrDcrServiceMissing	-258	No se dispone de un servicio externo al **CPE** de apoyo al **Cliente ECI** en las operaciones de desencriptación.
ErrDcrResourceMissing	-259	No se dispone de un recurso indefinido interno al **CPE** para el acceso o la desencriptación.
ErrDcrMmiMissing	-260	No está disponible el acceso del **Cliente ECI** al MMI.
ErrDcrDescrContinue	-261	El **Anfitrión ECI** sigue intentando desaleatorizar el contenido de este TS.
ErrDcrNetworkAccessCapability	-262	El **Anfitrión ECI** no tiene un recurso de acceso a la red que le permita localizar al TS requerido.
ErrDcrNetworkAccessResource	-263	El **Anfitrión ECI** no puede adquirir el recurso de acceso a la red para acceder al TS requerido.
ErrDcrPrioOverride	-264	Una tarea de mayor prioridad en el CPE necesita los recursos del **Asa de Medios**, por lo que debe terminarse la sesión del **Asa de Medios**.
RFU	Otros	Reservado para uso futuro.

9.6.2.3 Desaleatoriación de datos de un flujo de transporte

9.6.2.3.1 Introducción

El **Anfitrión ECI** puede solicitar al **Cliente ECI** que realice una sesión de desaleatorización (de un tipo específico: en este caso el tipo de difusión mpeg) para lo que le proporciona un **Asa de Medios** (véase la cláusula 9.1.2). El **Anfitrión ECI** proporcionará los datos de seguridad que especifique el **Cliente ECI** para desaleatorizar los datos.

Para desaleatorizar contenidos en la mayoría de los formatos del flujo de transporte, la **ECI** utiliza un modelo de temporización implícito para la sincronización de las palabras de control con el contenido ofrecido al desaleatorizador. En este modelo, el **Anfitrión ECI** proporciona al **Cliente ECI** datos de control de seguridad del flujo de transporte cuando éste es demultiplexado y desaleatorizado. El **Cliente ECI** proporciona en el momento adecuado las palabras de control necesarias (normalmente dos por cada flujo elemental, a menudo idénticas para todos los flujos elementales). Normalmente el **Cliente ECI** decodifica un ECM en forma de CW y carga inmediatamente dichas CW en el desaleatorizador. La aplicación de dichas palabras de control se sincroniza con el flujo mediante la señalización del contenido del flujo utilizando los bits de control de aleatorización a nivel de paquete TS o a nivel de paquete PES.

La partición de la API se realiza según se expone en las cláusulas siguientes:

1) Inicio, reinicio y detención de la desencriptación del flujo de transporte (cláusula 9.6.2.3).
2) Adquisición de datos de seguridad (cláusula 9.6.2.3.5).
3) Funciones de sintonización para la difusión (cláusula 9.6.2.3.6).

9.6.2.3.2 Formato del flujo de control y versión de sesiones

Los flujos de transporte desleatorizados en el contexto de un **Asa de Medios** con el tipo de sesión de medios **MhDvbTsBroadcast** cumplirán las especificaciones siguientes: [ISO/CEI 13818-1-1] (específicamente la aplicación de los bits de control de aleatorización a los paquetes TS) y [ETSI ETR 289].

9.6.2.3.3 Requisitos del procesamiento del Anfitrión ECI

9.6.2.3.3.1 Detección del cifrador de aleatorización

El **Anfitrión ECI** señalizará el modo de cifrado aplicable al **Cliente ECI** en base a las normas siguientes:

1) Para flujos DVB utilizará la señalización mediante el descriptor de aleatorización de la PMT tal como se define en [ETSI TS 103 127] y [ETSI TS 100 289].

2) Si no se encuentra descriptor alguno según el párrafo anterior y la fuente es una red de difusión DVB, el **Anfitrión ECI** asumirá que se utiliza CSA1, tal como se especifica en la definición del descriptor de aleatorización.

9.6.2.3.3.2 Detección de la identificación del CA

El **Anfitrión ECI** utilizará las siguientes secuencias de reglas de adquisición al objeto de establecer la lista de los ID de CA DVB aplicables a un servicio aleatorizado, cuando la aleatorización se detecta mediante bits de aleatorización de paquetes TS o PES en un flujo de transporte (originado en una red de difusión o de otra forma):

1) Intentará obtener los CA_descriptors transportados en la PMT del servicio. En caso de que no tenga éxito y el contenido es aleatorizado.

2) Intentará obtener el conjunto de CA_system_ids transportados en el descriptor de identificador del CA que transporta cualquier paquete de programas DVB, tabla SDT o tabla EIT aplicable para el contenido en cuestión.

NOTA – Para algunas fuentes de contenido basadas en flujos de transporte, el ID de CA o DRM aplicable puede conocerse por otros medios.

9.6.2.3.4 Inicio y detención de la desencriptación del flujo de transporte

9.6.2.3.4.1 Generalidades

El **Anfitrión ECI** puede iniciar la desencriptación del contenido en un **Asa de Medios** abierto utilizando recursos reservados del **Cliente ECI**. El **Anfitrión ECI** proporcionará una tabla "CA-PMT" con la especificación de los flujos elementales a desencriptar. En el Cuadro 9.6.2.3.4.1-1 figuran los mensajes API de desencriptación disponibles.

Cuadro 9.6.2.3.4.1-1 – API de desencriptación de contenido de un TS del Asa de Medios

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqDcrTsDescrStart	A	H→C	0x08	Solicita al **Cliente ECI** que desaleatorice o devuelva el estado desaleatorización de un programa en un TS.
reqDcrTsDescrStop	A	H→C	0x09	El **Anfitrión ECI** solicita el **Cliente ECI** que desaleatorice un **Asa de Medios**.
reqDcrTsDescrQuit	A	C→H	0x0A	El **Cliente ECI** termina una sesión de desaleatorización con el **Anfitrión ECI**.

9.6.2.3.4.2 Mensaje reqDcrTsDescrStart

H→C reqDcrTsDescrStart(ushort **mH**, uint **caPmtLen**, byte **caPmt[]) →**
C→H resDcrTsDescrStart(ushort **mH**, unit **sizeofEsStat**, descrStat **esStat[])**

- Este mensaje permite solicitar al **Cliente ECI** que comience la desencriptación de un programa tal como define **caPmt** en el flujo identificado por el **mH** o consulta en relación con la capacidad o las condiciones para ello.

Definición de los parámetros de la Petición:

mH: ushort	**Distintivo de medios** del flujo TS.
caPmtLen: uint	Longitud en bytes del parámetro **caPmt**.
caPmt: byte[]	El objeto ca_pmt se define en la cláusula 8.4.3 de [ETSI TR 101 202] en el orden de bytes de la red, con una interpretación modificada de los parámetros ca_pmt_list_management y ca_pmt_cmd_id tal como se define en el Cuadro 9.6.2.3.4.2-1.

Los valores y la semántica del parámetro ca_pmt_list_management serán conformes con las definiciones del Cuadro 9.6.2.3.4.2-1.

Cuadro 9.6.2.3.4.2-1 – Valores de ca_pmt_list_management

Nombre	Valor	Descripción
DcrTsDescrStartOnly	0x03	En el servicio debe desaleatorizarse un único programa. Puede ser un valor nuevo o actualizado.
DcrTsDescrStartUpdate	0x05	Mismo significado que **DcrTsDescrStartOnly**.
RFU	Otros	Reservado para uso futuro.

Los valores del parámetro ca_pmt_cmd_id serán idénticos a lo indicado en la cláusula 8.4.3 de [CEN EN 50221], con las restricciones siguientes:

1) El valor 0x02 (ok_mmi) no está permitido.

2) Los valores 0x01 (ok_scrambling) y 0x03 (query) no tendrán lugar en la misma estructura ca_pmt. Es decir, una **Petición** será una consulta pura o una petición de desaleatorización pura.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.
sizeofEsStat: uint	Número de bytes del parámetro esStat.
esStat: descrStat	Estado de desaleatorización de los flujos elementales, tal como se especifica en el parámetro **caPmt** de la **Petición**. En el Cuadro 9.6.2.3.4.2-2 se define descrStat. Un valor **descrStat.pid** solo se producirá una vez se haya alcanzado **esStat**. Cada parámetro elementary_PID de la estructura ca_pmt de [CEN EN 50221] solo ocurrirá una vez salvo que su correspondiente ca_pmd_cmd_id sea 0x04 (not_selected) en cuyo caso no ocurrirá en esStat.

Cuadro 9.6.2.3.4.2-2 – Definición de tipos de la estructura descrStat

```
typedef struct descrStat {
   ushort pid;
   uchar     caStatus
} descrStat;
```

pid: ushort	Valor del PID del flujo a desaleatorizar.
caStatus: uchar	Los valores corresponderán a la definición del parámetro CA_enable del objeto ca_pmt_reply en la cláusula 8.4.3 de [CEN EN 50221].

Información de la Semántica:

1) El **Anfitrión ECI** generará esta instrucción en caso de que deba modificarse el conjunto de flujos elementales a decodificar.

2) Si la sesión de medios se detiene, el **Anfitrión ECI** generará una **Petición reqDcrTsDescrEnd**. Si se produce un fallo al respecto, puede confundirse al **Cliente ECI** en relación al actual registro de consumo de contenidos del **Usuario** y los cargos asociados.

3) Los códigos de error conexos se definen en el Cuadro 9.6.2.3.4.2-3.

Precondiciones a la Petición:

1) **mH** está abierto y tiene un formato de TS.

Postcondiciones a la Petición:

1) El **Cliente ECI** puede iniciar actuaciones de desaleatorización y utilizar otras funciones conexas del TS del mH.

Cuadro 9.6.2.3.4.2-3 – Códigos de error de reqDcrTsStart

Nombre	Descripción
ErrDcrUserDelay	Véase el Cuadro 9.6.2.3.7-1.
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.3.4.3 Mensaje reqDcrTsDescrStop

H→C reqDcrTsDescrStop(ushort **mH) →**
C→H resDcrDescrStop(ushort **mH)**

- Este mensaje permite al **Anfitrión ECI** indicar al **Cliente ECI** que deberá detener la operación de desaleatorización del TS relacionado con el **mH** actual.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Precondiciones a la Contestación:

1) Se finaliza cualquier operación del **Cliente ECI** relacionada con la desaleatorización de **mH**.

9.6.2.3.4.4 Mensaje reqDcrTsDescrQuit

C→H reqDcrTsDescrQuit(ushort **mH**, **ushort** reason**) →**
H→C resDcrDescrQuit(ushort **mH)**

- Este mensaje permite al **Cliente ECI** informar al **Anfitrión ECI** que ha detenido el procesamiento de claves para la operación de desaleatorización del TS relacionada con el **mH** actual.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.
reason: ushort	El motivo por el cual el **Cliente ECI** ha finalizado el procesamiento de claves para la operación de desaleatorización se define en el Cuadro 9.7.2.5.9-1.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Precondiciones a la Contestación:

1) Se han finalizado todas las actividades del **Anfitrión ECI** relacionadas con la desaleatorización de **mH** o bien se devuelve un error.

Postcondiciones a la Contestación:

1) Se finalizarán inmediatamente todas las actividades del **Cliente ECI** relacionadas con el **mH** o bien se ha devuelto un error.

Cuadro 9.6.2.3.4.4-1 – Códigos de error de reqDcrTsDescrQuit

Nombre	Descripción
ErrDcrDescrContinue	Véase el Cuadro 9.6.2.3.7-1.

9.6.2.3.5 Adquisición de datos de desencriptación del Cliente ECI en el TS

9.6.2.3.5.1 Generalidades

El **Cliente ECI** puede adquirir datos del TS dentro de banda con fines de desencriptación en forma de secciones del flujo de transporte asociado con un **Asa de Medios**. La forma más sencilla de hacerlo es fijar los valores de un filtro de sección. Para acelerar la adquisición de los cambios que se produzcan en el canal puede fijarse un filtro de sección por defecto que incluya la PMT y el flujo ECM. También puede leer otras tablas MPEG y DVB normalizadas del **Anfitrión ECI**. Las secciones MPEG son estructuras de datos definidas en la cláusula 2.4.4.1 de [ISO/CEI 13818-1-1], estructura private_section(). Las funciones de esta parte de la API de TS MEPG figuran en el Cuadro 9.6.2.3.5.1-1.

Cuadro 9.6.2.3.5.1-1 – Mensajes de control de desaleatorización de TS del Anfitrión ECI

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setDcrTsSectionAcqDefault	set	C→H	0x10	Fija un filtro por defecto para la adquisición de una sección.
setDcrTsSectionAcq	set	C→H	0x11	Fija un filtro para adquisición de secciones.
reqDcrTsSection	A	H→C	0x12	Envía una sección adquirida al **Cliente ECI**.
reqDcrTsTable	A	C→H	0x13	El **Cliente ECI** adquiere una tabla del flujo.

9.6.2.3.5.2 Especificación del filtro de sección

Las secciones MPEG definidas en la cláusula 2.4.4.11 de [ISO/CEI 13818-1-1], pueden extraerse con arreglo a los especificado de un flujo de transporte entre un **Cliente ECI** al **Anfitrión ECI**. Un **Anfitrión ECI** permite hasta 8 filtros de sección para un **Cliente ECI**. Al establecer un filtro de sección el **Cliente ECI** puede filtrar un PID en el flujo TS con un número limitado de especificadores indirectos (por ejemplo, para PMT). Asimismo, permite al **Cliente ECI** establecer filtros positivos (campos de sección seleccionados que se corresponden con la especificación del **Cliente ECI**) y filtros negativos (datos de sección que difieren de la especificación del **Cliente ECI**). Las secciones filtradas pueden agruparse y transmitirse cuando se alcanza el tamaño máximo de la memoria intermedia o bien tan pronto como se adquieren.

El filtrado de bytes de sección no tendrá en cuenta los bytes segundo y tercero de una sección.

El Cuadro 9.6.2.3.5.2-1 presenta la especificación de un filtro de sección.

Cuadro 9.6.2.3.5.2-1 – Definición del tipo para 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 semántica es la siguiente:

pid: ushort	PID de los paquetes TS a filtrar. Los valores PID serán sus valores de 13 bits sin signo: es decir, entre 0x0000 y 0x1FFF. El PID de la PMT del flujo a adquirir se representa mediante 0x8000. El PID de un flujo ECM asociado a adquirir se representa mediante 0x8001.
cald: ushort	Este campo solo es pertinente si el valor del campo **pid** es 0x8001. En ese caso, el valor de este campo es el ID de CA MPEG/DVB del sistema de acceso condicional para el que se adquirirá el flujo ECM. El **Anfitrión ECI** analizará la PMT del servicio que debe desaleatorizar y determinará la concordancia entre el campo **cald** y los CA_descriptors (tal como se define en [ISO/CEI 13818-1-1]) aplicables al PID del video en caso de que esté presente, o el primer ES en la PMT, y utilizará el campo CA-PID en el descriptor de concordancia a fin de identificar el flujo ECM a adquirir y filtrar.
bufferSize: ushort	Tamaño máximo de la memoria intermedia. En dicha memoria se almacenará como mínimo una sección. Si este campo se pone a cero cada sección se transmitirá por separado.
timeout: uint	Temporización en ms para el filtrado de una sección. Se reinicia con cada sección filtrada con éxito. El valor cero significa que no existe temporización.
modeFlags: uint	Cuando se pone a cero el bit 0, el **Anfitrión ECI** no enviará dos veces la misma sección al **Cliente ECI**. A tal fin, el **Anfitrión ECI** utilizará una memoria intermedia de 64 kB como máximo con las secciones previamente adquiridas. Los restantes bits quedan en reserva y el **Cliente ECI** los pondrá a cero.
filter: byte []	Valor a comparar con los correspondientes bytes de sección.
mask: byte[]	Si un bit se pone a cero, se ignorará la concordancia con el valor de la sección.
neg: byte []	Si un bit se poner a uno, la concordancia con el bit de sección es negativa.

Existe una correspondencia entre una sección y un filtro cuando todos los bits de sección enmascarados con filtrado positivo concuerdan con su correspondiente valor del filtro y ningún bit de sección enmascarado con filtrado negativo concuerda con su correspondiente valor del filtro (en la hipótesis que existe al menos un bit cuyo filtrado ha sido negativo). La función **sectionFilterMatch** define una concordancia de sección (representada por **data** para los bytes de sección 1 y 3-18).

```
bool sectionFilterMatch(byte *data, *filter, *mask, *neg) {
    int i;
    bool posMatch, negMatch;

    posMatch = True;
    negMatch = True;

    /* si todos los bytes negativos son 0; siempre se cumplimenta el filtro negativo */
    for (i=0; i< DcrSectionFilterMaxlen; i++)
        negMatch &&= neg[i] == 0;

    /* establece la concordancia de los datos de la sección con criterios de filtrado
positivos y negativos */
    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 Mensaje reqDcrTsSectionAcqDefault

C→H setDcrTsSectionAcqDefault(ushort **mH**, uchar **filterNr**, dcrSectionFilterSpec **sectionFilter)**

- Este mensaje fija los filtros de sección por defecto que utilizará el **Anfitrión ECI** para adquirir información del flujo para el **Cliente ECI** una vez recibido el mensaje resDcrTsDescrStart. Esta función puede, por ejemplo, utilizarla el **Cliente ECI** para acelerar la adquisición de secciones de los ECM por el **Anfitrión ECI** durante un cambio de canal.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS en el que fijar el filtro de sección por defecto.
filterNr: uchar	Número del filtro a programar. El valor estará comprendido entre 0 y 7.
sectionFilter: dcrSectionFilterSpec	Especificación del filtro de sección en función de la sección, dcrSectionFilterSpec, cláusula 9.6.2.3.5.2.

Condición posterior:

- El **Anfitrión ECI** activará este filtro de sección inmediatamente después de la recepción satisfactoria de **resDcrTsDescrStart**. El **Anfitrión ECI** debe anticipar un **resDcrTsDescrStart** exitoso si ello es razonablemente posible.

9.6.2.3.5.4 Mensaje reqDcrTsSectionAcq

C→H setDcrTsSectionAcq(ushort **mH**, uchar **filterNr**, dcrSectionFilterSpec **sectionFilter)**

- Este mensaje fija los filtros de sección que utilizará el **Anfitrión ECI** para adquirir información del flujo **mH** para el **Cliente ECI**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS en el que fijar el filtro de sección por defecto.
filterNr: uchar	Número del filtro a programar. El valor estará comprendido entre 0 y 7.
sectionFilter: dcrSectionFilterSpec	Especificación del filtro de sección en función de la sección, dcrSectionFilterSpec, cláusula 9.6.2.3.5.2.

Información de la Semántica:

- La utilización de este mensaje después de fijar un filtro de sección por defecto modificará el filtro de sección hasta la emisión de la siguiente resDcrTsDescrStart en el mismo **Asa de Medios**, que lo reinicializará al valor del filtro de sección por defecto (si se ha establecido un valor por defecto).

Fijación posterior:

- El **Anfitrión ECI** fijará este filtro de sección.

9.6.2.3.5.5 Mensaje reqDcrTsSection

H→C reqDcrTsSection(ushort **mH**, uchar **filterNr**, uint **sectionDataLen**, byte **sectionData[]) →**
C→H resDcrTsSectionAcq (ushort **mH**, uchar **filterNr)**

- Este mensaje envía al **Cliente ECI** una o más secciones adquiridas por el **Anfitrión ECI** en el contexto del flujo TS identificado por el **mH** y el filtro identificado por **filterNr**.
- Los códigos de error conexos se definen en el Cuadro 9.6.2.3.5.5-1.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS en el que se fija el filtro de sección por defecto.
filterNr: uchar	Número del filtro a programar. El valor estará comprendido entre 0 y 7.
sectionDataLen: uint	Número de bytes de **sectionData**.
sectionData: byte []	La secuencia de private_sections (bytes en el orden de red) se define en [ISO/CEI 13818-1-1] sección 2.4.4.11. Cualquier sección con un error CRC no se transfiere al **Cliente ECI**.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.
filterNr: uchar	Número del filtro que ha sido programado.

Precondiciones a la Petición:

1) Las secciones habrán sido adquiridas por el **Anfitrión ECI** con arreglo a la especificación del filtro de sección o bien ha expirado la temporización del filtro.

2) Se acusa recibo del mensaje **reqDcrTsSection** anterior con **resDcrTsSection**.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** puede enviar el siguiente mensaje **reqDcrTsSection** desde el mismo filtro.

Cuadro 9.6.2.3.5.5-1 – Códigos de error de reqDcrTsSection

Nombre	Descripción
ErrDcrTsSectionTimeout	Véase el Cuadro 9.6.2.3.7-1.
ErrDcrTsSectionCrcErr	

9.6.2.3.5.6 Mensaje reqDcrTsTable

C→H reqDcrTsTable(ushort **mH**, uchar **tableId**, uint **timeout**, uint **maxLen)**
H→C resDcrTsTable(ushort **mH**, uint **TableDataLen**, byte **TableData[])**

- Este mensaje solicita al **Anfitrión ECI** que envíe las secciones que componen una tabla o subtabla estándar aplicables al programa que se desaleatoriza en el **mH**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS en el que se fija el filtro de sección por defecto.
TableId: uchar	Número del filtro a programar. Los valores válidos figuran en el Cuadro 9.6.2.3.5.6-1.
timeout: uint	Temporización en milisegundos. El valor 0 significa que no existe temporización.
maxLen: uint	Número máximo de bytes sectionData a devolver. El **Anfitrión ECI** hará un redondeo a la baja hasta el número de secciones más elevado dentro de este límite.

Cuadro 9.6.2.3.5.6-1 – Valores de ca_pmt_list_management

Nombre	Valor	Descripción
DcrTsTableMpegPat	0x0000	Tabla PAT conforme a [ISO/CEI 13818-1-1].
DcrTsTableMpegCat	0x0001	Tabla CAT conforme a [ISO/CEI 13818-1-1].
DcrTsTableMpegPmt	0x0002	Tabla PMT del programa seleccionado conforme a [ISO/CEI 13818-1-1]. Si la aplicación utiliza una PMT compuesta el resultado es vacío.
DcrTsTableDvbNit	0x0140	Tabla NIT de la red de suministro real tal como se especifica en [ETSI EN 300 468] y [ETSI TS 101 211]. En redes de cable que utilicen NIT_{other} para transportar tablas asociadas a las regiones de dicha red se designará la tabla NIT_{other} aplicable a la región del **CPE**.
DcrTsTableDvbSdt	0x0142	Tabla SDTactual_current especificada en [ETSI EN 300 468] y [ETSI TS 101 211].

Cuadro 9.6.2.3.5.6-1 – Valores de ca_pmt_list_management

Nombre	Valor	Descripción
DcrTsTableDvbBat	0x014A	Tabla BATactual según se especifica en [ETSI EN 300 468] para el paquete de programas utilizado activamente por el **Anfitrión ECI** y/o su aplicación.
DcrTsTableDvbEitPf	0x014E	Tabla EIT actual y siguiente según se especifica en [ETSI EN 300 468] y [ETSI TS 101 211].
DcrTsDescrStartUpdate	0x05	Mismo significado que **DcrTsDescrStartOnly**.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.
TableDataLen: uint	Número de bytes en TableData.
TableData: byte []	La secuencia de private_sections (bytes en orden de red) que representa la (sub)tabla se define en la sección 2.4.4.11 [ISO/CEI 13818-1-1].

Información de la Semántica:

- El **Anfitrión ECI** utilizará filtros de sección para adquirir nuevos datos para todas las tablas que pueda solicitar el **Cliente ECI** (así como para otros fines de este). El **Anfitrión ECI** enviará una vez las secciones de las tablas. El **Anfitrión ECI** detendrá la **Contestación** si precisa adquirir la tabla solicitada. La tabla será "actualizada" y utilizará los últimos datos completos a disposición del **Anfitrión ECI**. Los códigos de error se definirán en el Cuadro 9.6.2.3.5.6-2.

NOTA – Una tabla siempre puede ser sustituida más adelante por una versión posterior incluida en un tren.

  – Las tasas mínimas de repetición para la actualización de tablas SI DVB pertinentes se definen en [b-UIT-T J Supl. 7].
  – PAT, CAT y PMT: los datos existen desde hace más de 20 segundos.

Cuadro 9.6.2.3.5.6-2 – Códigos de error de reqDcrTsTable

Nombre	Descripción
ErrDcrTsSectionTimeout	Véase el Cuadro 9.6.2.3.7-1.
ErrDcrTsSectionCrcErr	

9.6.2.3.6 Control de la fuente del Cliente ECI

9.6.2.3.6.1 Generalidades

Un **Cliente ECI** tiene la capacidad de leer el tipo de fuente del flujo de transporte, controlar (redireccionar) la fuente del flujo de transporte y redireccionar el programa y/o los componentes decodificados por el **Anfitrión ECI**. Los mensajes figuran en el Cuadro 9.6.2.3.6.1-1.

Cuadro 9.6.2.3.6.1-1 – Mensajes de la API de control de la fuente del cliente TS

Mensaje	Tipo	Dir.	Etiqueta	Descripción
getDcrTsSource	get	C→H	0x18	El **Cliente ECI** obtiene la fuente del TS.
reqDcrTsRelocate	A	C→H	0x19	Los **Clientes ECI** reubican la fuente del TS.
reqDcrTsSelectPrg	A	C→H	0x1A	El **Cliente ECI** selecciona el programa en el TS mediante el número de programa.
reqDcrTsSelectPmt	A	C→H	0x1B	Selecciona el programa en el TS mediante la PMT
reqDcrTsSelectCancel	A	C→H	0x1C	El **Cliente ECI** cancela su selección de programa anterior.

9.6.2.3.6.2 Mensaje getDcrTsSource

C→H tsSourceType **getDcrTsSource(**ushort **mH)**

- Este mensaje devuelve el tipo de fuente del **Asa de Medios** en términos de tipo de red y localizador en la red.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del flujo TS para obtener el tipo y localización del flujo sintonizado.

Definición de propiedades:

Las definiciones de las propiedades se presentan en el Cuadro 9.6.2.3.6.2-1.

Cuadro 9.6.2.3.6.2-1 – Definición de tipos de la estructura tsSourceType

```
#define MaxTsSourceDescr 254

typedef struct tsSourceType{
    ushort tsSourceTag ;
    byte   tsSourceDescr[MaxTsSourceDescr] ;
} tsSourceType ;
```

tsSourceTag: ushort	Tipo de la fuente de TS. Los valores definidos se presentan más abajo, incluido el significado de **tsSourceDescr**.
tsSourceDescr: byte[MaxTsSourceDescr]	El significado depende de tsSourceTag tal como figura en el Cuadro 9.6.2.3.6.2-2.

Cuadro 9.6.2.3.6.2-2 – Significado de la etiqueta tsSource

Nombre	Valor	Descripción
tsSourceDvbTuner	0x0001	La fuente del TS es un sintonizador DVB. El tsSourceDescr contiene un único descriptor del Cuadro 9.6.2.3.6.2-3 en el orden de bytes de la red.
tsSourceDvbFile	0x0002	La fuente del TS es un fichero u otro activo no sintonizable como por ejemplo una red IP (véase [b-ETSI TS 102 034]). El campo tsSourceDescr no está definido.
tsDvbDuplet	0x8003	La fuente o el TS pueden encontrarse en la red actual utilizando el ID de red y el ID del flujo de transporte originales. El tsSourceDescr contendrá el orden de los bytes de red de `struct dvbDuplet {ushort onid; ushort tsid};` El mensaje getDcrTsSource no devolverá este valor (en su lugar devolverá un tsSourceDvbTuner) pero puede ser utilizado en un mensaje reqDcrTsRelocate.
RFU	Otros	Reservado para uso futuro.

Los valores mayores que 0x7FFF no son localizadores absolutos y no serán devueltos por **getDcrTsSource**.

Cuadro 9.6.2.3.6.2-3 – Descriptores de la fuente del sintonizador DVB

Nombre del descriptor del servicio DVB	Valor de la etiqueta del descriptor DVB
terrestrial_delivery_system_descriptor (descriptor del sistema de distribución terrestre)	0x5A
T2_delivery_system_descriptor (descriptor del sistema de distribución T2)	0x7F, 0x04
satellite_delivery_system_descriptor (descriptor del sistema de distribución por satélite)	0x43
S2_delivery_system_descriptor (descriptor del sistema de distribución S2)	0x79
cable_delivery_system_descriptor (descriptor del sistema de distribución por cable)	0x44
C2_delivery_system_descriptor (descriptor del sistema de distribución C2)	0x7F, 0x0D

Los descriptores se utilizarán como se define en [ETSI EN 300 468], y contendrán una única frecuencia de destino.

9.6.2.3.6.3 Mensaje reqDcrTsRelocate

C→H reqDcrTsRelocate(ushort **mH**, tsSourceType **tsLoc) →**
H→C resDcrTsRelocate(ushort **mH)**

- Este mensaje solicita al **Anfitrión ECI** que reubique la fuente del TS en **tsLoc**. Los códigos de error conexos se definen en el Cuadro 9.6.2.3.6.3-1.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS para la reubicación/resintonización.
tsLoc: tsSourceType	En el Cuadro 9.6.2.3.6.2-1 se define la ubicación donde recolocar el flujo.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS que fue reubicado.

Información de la Semántica:

- Si es necesario disponer de un recurso de acceso a la red (por ejemplo, sintonizador/demodulador para difusión) distinto del actualmente asignado al **Asa de Medios**, es posible que el **Anfitrión ECI** no pueda acceder a la **Petición** por las limitaciones de recursos existentes.
- Cuando se consigue un resintonización satisfactoria se finaliza cualquier filtrado o desaleatorización en curso. La adquisición por defecto comenzará una vez que se haya adquirido el TS.

Cuadro 9.6.2.3.6.3-1 – Códigos de error de reqDcrTsRelocate

Nombre	Descripción
ErrDcrTsNetworkAccessCapability	Véase el Cuadro 9.6.2.3.7-1.
ErrDcrTsNetworkAccessResource	
ErrDcrTsNetworkAccessFail	

9.6.2.3.6.4 Mensaje reqDcrTsSelectPrg

C→H reqDcrTsSelectPrg(ushort **mH**, ushort **prgNumber) →**
H→C resDcrTsSelectPrg(ushort **mH)**

- Este mensaje establece que el programa que el **Anfitrión ECI** debe seleccionar para su desaleatorización corresponde a **prgNumber**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS
prgNumber: ushort	Número de programa en las tablas PAT y PMT MPEG (véase [ISO/CEI 13818-1-1]) en el TS que define el servicio que debe seleccionar el **Anfitrión ECI**.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Información de la Semántica:

- El **Anfitrión ECI** localizará la PAT en el TS indicado por **mH**. Localizará el PID de la PMT comparando **prgNumber** y program_number. Adquirirá la PMT del PID localizado y usará las funciones ordinarias del **Anfitrión ECI** para seleccionar los componentes del programa seleccionado para su presentación. Si todo ello finaliza con éxito, el **Anfitrión ECI** emitirá un **Petición reqDcrTsDescrStart** para iniciar la desaleatorización del programa.

Postcondiciones a la Petición:

1) Si el **Anfitrión ECI** se encontraba desaleatorizando un programa no seleccionado por una **Petición reqDcrTsSelectPrg** o **reqDcrTsSelectPmt Request,** almacenará los parámetros de selección del programa de forma que posteriormente pueda volver a ese mismo programa tras una **reqDcrTsSelectCancel**.

Postcondiciones a la Contestación:

1) Si no se devuelve error, el **Anfitrión ECI** enviará una **reqDcrTsDescrStart**.

Los códigos de error del mensaje petición de inicio de desencriptación se incluyen en el Cuadro 9.6.2.3.6.4-1.

Cuadro 9.6.2.3.6.4-1 – Códigos de error de reqDcrTsSelectPrg

Nombre	Descripción
ErrDcrTsPrgNumberNotInPsi	Véase el Cuadro 9.6.2.3.7-1.
ErrDcrTsComponentSelectError	

9.6.2.3.6.5 Mensaje reqDcrTsSelectPmt

C→H reqDcrTsSelectPmt(ushort **mH**, uint **pmtLen**, byte **pmt[]) →**
H→C resDcrTsSelectPmt(ushort **mH)**

- Este mensaje selecciona un nuevo programa que el **Anfitrión ECI** debe desaleatorizar mediante el envío de una tabla PMT MPEG en la que se definen los componentes del programa en el flujo de transporte que identifica **mH**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.
pmtLen: uint	Número de bytes del parámetro **pmt**.
pmt: byte	private_section que contiene una tabla PMT conforme con [ISO/CEI 13818-1-1].

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Información de la Semántica:

- Esta instrucción permite a un **Cliente ECI** seleccionar componentes de un TS que no tengan tablas PAT y PMT adecuadas. El **Anfitrión ECI** utilizará **pmt** para seleccionar los componentes del programa seleccionado para su presentación. Si todo ello finaliza con éxito, el **Anfitrión ECI** generará un **Petición reqDcrTsDescrStart** para iniciar la desaleatorización del programa.

Postcondiciones a la Petición:

1) Si el **Anfitrión ECI** se encontraba desaleatorizando un programa no seleccionado por una **Petición reqDcrTsSelectPrg** o **reqDcrTsSelectPmt**, almacenará los parámetros de selección del programa de forma que posteriormente pueda volver a ese mismo programa tras una **reqDcrTsSelectCancel**.

Postcondiciones a la Contestación:

1) Si no se devuelve un error, el **Anfitrión ECI** enviará una **reqDcrTsDescrStart**.

Los códigos de error de este mensaje API se muestran en el Cuadro 9.6.2.3.6.5-1.

Cuadro 9.6.2.3.6.5-1 – Códigos de error de reqDcrTsSelectPmt

Nombre	Descripción
ErrDcrTsComponentSelectError	Véase el Cuadro 9.6.2.3.7-1.

9.6.2.3.6.6 Mensaje reqDcrTsSelectCancel

C→H reqDcrTsSelectCancel(ushort **mH) →**
H→C resDcrTsSelectCancel(ushort **mH)**

- Este mensaje cancela una **reqDcrTsSelectPrg** o **reqDcrTsSelectPmt** previa del Cliente ECI, y vuelve al programa original seleccionado por el **Anfitrión ECI** en el TS que identifica **mH**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** puede enviar posteriormente una **reqDcrTsDescrStart** para retomar la desaleatorización del programa original.

9.6.2.3.7 Códigos de error de la API de sesión de medios para medios TS

Los valores y significados de errores específicos de la API que pueden devolver los mensajes de **Contestación** de la API figuran en el Cuadro 9.6.2.3.7-1.

Todas las peticiones de **Asa de Medios** específica del TS devuelven un código de error para el parámetro **Asa de Medios** si se aplican a un **Asa de Medios** que no sea del TS.

Cuadro 9.6.2.3.7-1 – Códigos de error de las API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrDcrTsUserDelay	-256	Se ha producido una demora prolongada en espera de información de entrada del **Usuario** necesaria para completar la operación. La operación no se ha completado.
ErrDcrTsCardMissing	-257	La **Tarjeta inteligente** necesaria para la sesión no está accesible/disponible.
ErrDcrTsServiceMissing	-258	No hay disponible un servicio externo al **CPE** necesario para soportar las operaciones de desencriptación del **Cliente ECI**.
ErrDcrTsResourceMissing	-259	No hay disponible un recurso indefinido interno al **CPE** necesario para acceder al contenido o para su desencriptación.
ErrDcrTsMmiMissing	-260	No está disponible el acceso del **Cliente ECI** al MMI.
ErrDcrDescrContinue	-261	El **Anfitrión ECI** continúa intentando desaleatorizar el contenido en este TS.
ErrDcrTsSectionTimeout	-262	Ha vencido una temporización para la adquisición de una sección.
ErrDcrTsSectionCrcErr	-263	Las secciones se han obtenido en el plazo del temporizador, aunque con errores CRC. Normalmente eso significa que el flujo está muy corrompido.
ErrDcrTsNetworkAccessCapability	-264	El **Anfitrión ECI** no dispone de un recurso de acceso a la red para localizar el TS solicitado.
ErrDcrTsNetworkAccessResource	-265	El **Anfitrión ECI** no puede adquirir el recurso de acceso a la red para acceder al TS solicitado.
ErrDcrTsNetworkAccessFail	-266	El recurso de acceso a la red fracasó en la adquisición (de manera fiable) del TS solicitado.
ErrDcrTsPrgNumberNotInPsi	-267	Sobre la base de la PAT no pudo localizarse una PMT con el correspondiente número de programa.

Cuadro 9.6.2.3.7-1 – Códigos de error de las API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrDcrTsComponentSelectError	-268	No pudo seleccionarse un componente de la PMT para demultiplexación/desaleatorización.
ErrDcrTsPidNotDescrambled	-269	El **Anfitrión ECI** no seleccionó un Pid para la desaleatorización.
ErrDcrTsCwIdNotValid	-270	Se ha hecho referencia a un ID inválido de palabra de control.
RFU	Otros	Reservado para uso futuro.

9.6.2.4 Desencriptación del contenido de ficheros y de flujos

9.6.2.4.1 Introducción

En esta cláusula se define una API del **Cliente ECI** /**Anfitrión ECI** que permite a un **CPE** y a las aplicaciones descargadas interactuar con un **Cliente ECI** de seguridad a través del **Anfitrión ECI** para desaleatorizar un contenido que tiene el formato ISOBMFF [ISO/CEI 23001-9] o cualquier otro fichero o flujo en el que el **Anfitrión ECI** (o el CPE subyacente o la aplicación descargada que actúa a través del mismo):

- pueda extraer los datos de control de seguridad requeridos y transferirlos al **Cliente ECI** del fichero o del flujo;
- permita aplicar correctamente las claves de desaleatorización generadas por el **Cliente ECI** (sincronizadas) al contenido mediante identificadores de clave (Key-ID).

Los ficheros ISOBMFF [ISO/CEI 23001-9] tienen un formato de empaquetamiento común para muchos métodos de descarga que no se ejecutan en tiempo real ni son adaptables. También existe un método de encriptación común definido para esos formatos de fichero: CENC [ISO/CEI 23001-7]. Asimismo, la norma relativa al formato de flujos adaptables MPEG-Dash [ISO/CEI 23009-1] y [ETSI TS 103 285] está basada en ISOBMFF, y distintos sistemas DRM (en algunas ocasiones preexistentes) utilizan sus propios subformatos ISOBMFF patentados (con identificador de "marca" de firma).

Una sección de la API permite al **Cliente ECI** especificar los datos que precisa del fichero ISOBMFF para poder realizar esa decodificación, permitiendo que aplicaciones DRM patentadas (no conformes con CENC) de ISOBMFF sean utilizadas por aplicaciones del **CPE**. Los aspectos específicos de la desaleatorización de muestras deberían ser gestionados por el **Anfitrión ECI:** es decir, pueden ser conformes con CENC o requerir extensiones patentadas en el **Anfitrión ECI**.

La API tiene las secciones siguientes:

1) Inicio y detención de la desaleatorización.
2) Establecimiento de los valores de los filtros de adquisición de datos de seguridad específicos del **Cliente ECI**.
3) API de la clave de desencriptación (palabra de control).

9.6.2.4.2 Especificaciones aplicables

Los ficheros ISOBMFF a los que se hace referencia en esta cláusula serán conformes con [ETSI TS 103 285]. Los ficheros ISOBMFF conformes con CENC (tal como exige el descifrado estándar) serán conformes con [ISO/CEI 23001-7].

Los datos de flujo conformes con Dash también serán conformes con [ISO/CEI 23009-1]. Los **Anfitriones ECI** que implementen Dash serán (como mínimo) conformes con [ISO/CEI 23001-7], [ISO/CEI 23001-9] y [ETSI TS 103 285] en la medida en que sean aplicables al alcance funcional del **CPE**.

9.6.2.4.3 Requisitos del procesamiento del Anfitrión ECI

9.6.2.4.3.1 Detección de la identificación del sistema de desencriptación

El **Anfitrión ECI** podrá adquirir la lista de sistemas de desencriptación aplicables del contenedor de contenidos sobre la base de las reglas siguientes:

1) Para todos los ficheros ISOBMFF y MP4, el **Anfitrión ECI** adquirirá la caja tipo de fichero ('ftyp', *file type* box) y la caja tipo de segmento ('styp', *segment type* box) y utilizará los campos `major_brand` y `compatible_brands[]` para establecer la correspondencia entre contenido y **Clientes ECI**.

2) Para ficheros codificados en CENC ISOBMFF, el **Anfitrión ECI** obtendrá las Cajas de encabezamiento específico del sistema de protección ('pssh', *protection system specific header*) de cualquiera de las posibles ubicaciones (véase [ISO/CEI 23001-7]) y obtendrá del campo `SystemID` las UIDD de los sistemas DRM adecuados para la desencriptación del contenido. Estos ficheros pueden reconocerse mediante una Caja información del esquema de protección ('sinf', *protection scheme information* box) que contiene la caja tipo de esquema ('schm' *scheme type* box) cuyo campo `scheme_type` sea igual a 'cenc' o 'cbc1' y la versión principal del campo `scheme_version` sea 0x0001. La definición y ubicación de las cajas 'sinf' se especifica en [ISO/CEI 23001-7].

3) En el caso del contenido MPEG-Dash, el **Anfitrión ECI** adquirirá todos los descriptores ContentProtection de la MPD que contengan un UUID específico (que comience por "urn:uuid:xxxxx", siendo xxxxx el UUID) para el atributo @SchemeIdUri a fin de establecer la concordancia con las UUID DRM del **Cliente ECI** o de contener un ID de sistema de acceso condicional con arreglo a [ETSI TS 103 285] en el atributo @value (véase [b-DASH-IF ID] para la definición de este identificador genérico). El **Anfitrión ECI** adquirirá todos los descriptores ContentProtection a fin de establecer la concordancia con las capacidades del **Cliente ECI**. Convertirá cualquier caja PSSH en la correspondiente representación binaria ISOBMFF.

El proceso para el establecimiento de correspondencias entre contenido y **Clientes ECI** se describe en la cláusula 9.6.2.4.5.2.1.

9.6.2.4.3.2 Detección del tipo de aleatorización

Los **Anfitriones ECI** señalizarán el modo desaleatorización aplicable al **Cliente ECI** sobre la base de las reglas siguientes:

1) Para ficheros codificados CENC ISOBMFF podrán aplicarse las reglas definidas en [ISO/CEI 23001-7] para detectar el cifrador (AES-CTR o AES-CBC) incluida la selección del byte que indica no aleatorizado/aleatorizado, el relleno y la extracción y aplicación del vector de inicialización tal como se define en [ISO/CEI 23001-7].

2) En el caso de contenido DASH MPEG en formato ISOBMFF, se aplicará AES-CTR (con rotación de clave) para la desaleatorización, tal como se define en [ETSI TS 103 285].

9.6.2.4.3.3 Filtrado de datos de seguridad del contenedor del contenido por defecto

El **Anfitrión ECI** transferirá cualquier caja del contenedor designado para el **Cliente ECI** que contenga información (opaca) cuando resulte pertinente para el proceso de desaleatorización. Específicamente, contiene espacio para las siguientes cajas CENC ISOBMFF y contenido Dash en formato ISOBMFF:

1) Para:

   a) Cajas de encabezamiento específico del sistema de protección ('pssh', *protection system specific header*) en cajas 'moov' y 'moof' que se correspondan con el UUID del ID del sistema DRM del **Cliente ECI**, pertinentes para su decodificación, ahora o en un futuro próximo.

b) Cajas de información del esquema de protección, 'sinf', si el **Cliente ECI** necesita acceder a cajas 'sinf'.

9.6.2.4.3.4 Desaleatorización de contenido

El **Anfitrión ECI** será responsable de interpretar el modo desaleatorización, identificar los datos a desaleatorizar y procesar los datos utilizando el desaleatorizador con los ID de clave adecuados para identificar las claves que pone a disposición el **Cliente ECI**.

Para que el **Cliente ECI** calcule las claves asociadas, el **Anfitrión ECI** transferirá los datos de control de seguridad necesarios desde el contenedor de contenidos al **Cliente ECI** a su debido tiempo.

9.6.2.4.4 API de sesión de medios para medios organizados en ficheros y en flujos

9.6.2.4.4.1 Generalidades

El **Anfitrión ECI** puede iniciar la desencriptación del contenido en un **Asa de Medios** utilizando los recursos del **Cliente ECI** reservados. El **Anfitrión ECI** proporcionará los datos de inicialización para que el **Cliente ECI** inicie la evaluación de los derechos de acceso.

Cuadro 9.6.2.4.4.1-1 – API de desencriptación de contenido del TS del Distintivo de medios

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqDcrFileStart	A	H→C	0x01	**Solicita** al **Cliente ECI** que desaleatorice o devuelva el estado de desaleatorización de un fichero o un flujo.
reqDcrFileStop	A	H→C	0x02	El **Anfitrión ECI** solicita al **Cliente ECI** que detenga el procesamiento de la clave para la operación de desaleatorización aplicada a un **Asa de Medios**.
reqDcrFileQuit	A	C→H	0x03	El **Cliente ECI** cancela una operación de desaleatorización con el **Anfitrión ECI**.

9.6.2.4.4.2 Mensaje reqDcrFileStart

H→C reqDcrFileStart(ushort **mH**, uchar **reqType**, uchar **dataType**, uint **initDataLen**, byte **initData[]) →**
C→H resDcrFileStart(ushort **mH**, uchar **dcrStat)**

- Este mensaje solicita al **Cliente ECI** que devuelva el estado de desaleatorización y/o inicie una sesión de desaleatorización del contenido asociado a mH. El **Anfitrión ECI** suministra los datos iniciales para que el **Cliente ECI** comience la adquisición y evaluación de una licencia conforme con el formato del contenedor/ encriptación.

Definición de los parámetros de la Petición:

mH: ushort	**Distintivo de medios** del fichero.
reqType: uchar	Tipo de **Petición** (inicio de desaleatorización o consulta sobre la licencia) definida en el Cuadro 9.6.2.4.4.2-1.
dataType: uchar	Tipo de InitData.
initDataLen: uint	Longitud en bytes del contenedor initData.
initData: byte	Datos de inicialización tomados del contenido tal como define dataType. La codificación de initDat se define en el Cuadro 9.6.2.4.4.2-2.

Cuadro 9.6.2.4.4.2-1 – Codificación de reqType

Nombre	Valor	Descripción
ReqTypeDcr	0x01	Inicio de la desaleatorización; dialoga con el **Usuario** si es necesario.
ReqTypeInq	0x02	Consulta las opciones de desaleatorización.
RFU	Otros	Reservado para uso futuro.

Cuadro 9.6.2.4.4.2-2 – Codificación de initData

dataType	Valor	Descripción
FmtIsoCenc	0x04	Cajas PSSH ISOBMFF (véase [ISO/CEI 23001-7]) identificadas que concuerdan con el ID de DRM en el MatchSpecifier del **Cliente ECI**.
FmtIsoCencDash	0x05	Cajas PSSH ISOBMFF (véase [ISO/CEI 23001-7]) en el MPD (véase [ISO/CEI 23007-1]) o en el segmento de inicialización (véase [ISO/CEI 23009-1]) que concuerdan con el ID de DRM en el MatchSpecifier del **Cliente ECI**.
FmtIsoProp	0x06	El **Anfitrión ECI** puede transferir datos al **Cliente ECI** sobre la base de conocimientos patentado. El **Cliente ECI** podrá interpretar estos datos sobre la base del mismo conocimiento patentado común.
FmtIsoPropDash	0x07	Incluye FmtIsoProp, la indicación de que los datos son una fuente DASH.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.
dcrStat: uchar	Estado de desaleatorización; véase el Cuadro 9.6.2.4.4.2-3.

Cuadro 9.6.2.4.4.2-3 – Estado de desaleatorización

Nombre	Valor	Descripción
DcrStatNo	0x00	La desaleatorización no es posible (el sistema DRM no tiene capacidad de desaleatorización).
DcrStatOk	0x01	Se inicia la desaleatorización; si es necesario, inicia el diálogo con el **Usuario**.
DcrStatDialog	0x02	Es necesario dialogar con el **Usuario**.
DcrStatPay	0x03	Es necesario efectuar el pago, posiblemente también dialogar con el **Usuario**.
DcrStatDrmNok	0xFE	El sistema DRM no tiene la capacidad de desaleatorizar este contenido.
RFU	Otros	Reservado para uso futuro.

Información de la Semántica:

- Basado en las consultas el **Cliente ECI** no iniciará diálogos de **Usuario,** sino que evaluará la capacidad de desaleatorizar el contenido desencriptando las condiciones de la licencia con el servidor de licencias sin entablar un diálogo con el **Usuario**.

Precondiciones a la Petición:

1) A la espera del **Asa de Medios**.

Precondiciones a la Contestación:

1) Si el **Cliente ECI** puede desaleatorizar el contenido y reqType es OK, el **Cliente ECI** estará preparado para generar claves de desaleatorización.

Los códigos de error para el mensaje de petición de inicio de desencriptación figuran en el Cuadro 9.6.2.4.4.2-4.

Cuadro 9.6.2.4.4.2-4 – Códigos de error de reqDcrFileStart

Nombre	Descripción
ErrDcrFileUserDelay	Véase el Cuadro 9.6.2.4.7-1.
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	

9.6.2.4.4.3 Mensaje reqDcrFileStop

H→C reqDcrFile Stop(ushort **mH) →**
C→H resDcrFile Stop(ushort **mH)**

- Este mensaje permite al **Anfitrión ECI** detener la desencriptación de ficheros.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del fichero.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del fichero.

Precondiciones a la Contestación:

1) El **Cliente ECI** ha terminado cualquier operación relacionada con la desencriptación de contenido.

9.6.2.4.4.4 Mensaje reqDcrFileQuit

C→H reqDcrFileQuit(ushort **mH**, uint **reason) →**
H→C resDcrFile Quit(ushort **mH)**

- Este mensaje permite al **Cliente ECI** informar al **Anfitrión ECI** que ha terminado el procesamiento de la clave para una operación de desencriptación de ficheros. Los códigos de error conexos se definen en el Cuadro 9.6.2.4.4.4-1.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.
reason: uint	Valores según se definen en el Cuadro 9.7.2.5.9-1.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del fichero.

Precondiciones a la Contestación:

1) Se dan por finalizadas todas las actividades del **Anfitrión ECI** relacionadas con la desaleatorización de **mH**, o bien, se devuelve un error.

Postcondiciones a la Contestación:

1) Se terminará inmediatamente cualquier actividad del **Cliente ECI** relacionada con **mH**, o bien, se devolverá un error.

Cuadro 9.6.2.4.4.4-1 – Códigos de error de reqDcrFileQuit

Nombre	Descripción
ErrDcrFileDescrContinue	Véase el Cuadro 9.6.2.4.7-1.

9.6.2.4.5 Adquisición de datos de seguridad específicos del Cliente ECI

9.6.2.4.5.1 Generalidades

El **Anfitrión ECI** realizará una adquisición de datos estándar con relación a los datos que deben decodificarse para obtener la información que el **Cliente ECI** necesita para el cálculo de la clave. El **Cliente ECI** puede indicar que se trata de una adquisición de datos específica que excede de los datos estándar que proporciona el **Anfitrión ECI**. El **Anfitrión ECI** mantendrá un número limitado de filtros para la adquisición de dichos datos.

Cuadro 9.6.2.4.5.1-1 – API de filtro de datos

reqDcrFileFilter	req	C→H	0x04	El **Cliente ECI** solicita al **Anfitrión ECI** que fije un filtro de datos para la adquisición de datos de seguridad.
reqDcrFileData	A	C→H	0x05	El **Cliente ECI** solicita al **Anfitrión ECI** que en la adquisición de datos utilice el Filtro de fichero.

9.6.2.4.5.2 Especificación del Filtro de fichero

9.6.2.4.5.2.1 Definición del Filtro de fichero genérico

La especificación del filtro de datos de fichero se basa en una especificación subyacente del formato del fichero. El filtro se define en el contexto de un formato de fichero definido. En el Cuadro 9.6.2.4.5.2.1-1 se define la especificación del Filtro de fichero genérico.

Cuadro 9.6.2.4.5.2.1-1 – Especificación del Filtro de fichero genérico

```
typedef struct dcrFileFilterSpec {
    ushort filterType;          // se define en el Cuadro 9.6.2.4.5.2.1-3
    ushort filterLen;
    byte   filter[filterLen]; // se formateará con arreglo al filterType
} dcrFileFilterSpec;
```

Cuadro 9.6.2.4.5.2.1-2 – Tipos de Filtro de fichero

FileFilterIsobmff	0x0001	El Filtro de fichero para datos con formato ISMBMFF se define en la cláusula 9.6.2.4.5.2.2.
RFU	Otros	Reservado para uso futuro.

9.6.2.4.5.2.2 Definición del Filtro de fichero específico ISOBMFF

La especificación del filtro para ficheros con formato ISOBMFF se define en el Cuadro 9.6.2.4.5.2.2-1.

Cuadro 9.6.2.4.5.2.2-1 – Especificación del Filtro de fichero ISOBMFF

```
#define MaxFilterFile 16 // número máximo de bytes de la caja que se filtran
#define MaxContainers  4 // número máximo de cajas contenedoras de una caja
#define MaxUuidLen    16 // Longitud en bytes de un UUID

typedef struct BoxSpec {
    uint   boxType                  // código 4CC del tipo de caja
    byte   extendedType[MaxUuidLen]
                                    // UUID para boxType=='uuid', en otro caso carece de
                                    significado
    byte   filter[MaxFileFilter];    // concordará con los bytes de la caja siguiente
    byte   filterMask[MaxFilter];
    ushort dataLen;                 // cantidad máxima de datos de caja a adquirir
} 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;
}
```

El **Anfitrión ECI** analizará el fichero y adquirirá cajas (boxes) que mantengan una correspondencia con el campo **box** y que a su vez estén contenidas en cajas que concuerden con cualquier matriz **contenedora**. El **Anfitrión ECI** no tendrá en cuenta cajas no definidas en [ISO/CEI 14496-12] o [ISO/CEI 23001-7].

El **boxType** del campo **contenedor** de **dcrFileFilterIsobmff** puede fijarse a '****' para indicar que se trata de un comodín. En ese caso, los demás campos del **contenedor** no tendrán significado alguno y se pondrán a 0 para reflejar que no existe correspondencia.

Los campos **filter** (filtro**)** y **filterMask** (máscara**)** de **BoxSpec** se compararán con los primeros bytes después del campo tipo de una caja a procesar. Para "cajas completas" ("full boxes", véase [ISO/CEI 14496-12]) se trata del campo versión y bandera. La comparación se establecerá con arreglo a la función boxMatch, asignando el parámetro **boxLen** al número de bytes que siguen a boxtype y extended_type de la caja, el parámetro **boxData** al comienzo de esos bytes, el parámetro **filter** al campo **boxSpec.filter** y el parámetro **filterMask** al valor del campo **boxSpec.filterMask**.

Los datos que devuelve el filtro son las cajas (en secuencia) que concuerdan con el filtro tras el análisis del filtro por el **Anfitrión ECI**. El **Anfitrión ECI** puede agrupar las cajas a su conveniencia, pero no debe demorar de forma innecesaria la transferencia de las cajas al **Cliente ECI** ya que ello puede impedir que el **Cliente ECI** genere las claves de desaleatorización necesarias.

9.6.2.4.5.2.3 Mensaje reqDcrFileFilter

C→H setDrcFileFilter(ushort **mH**, uchar **filterNr**, dcrFilleFilterSpec ***dataFilter)**

- Este mensaje solicita al **Anfitrión ECI** que fije un filtro de datos basado en el dataFilter para la adquisición de datos de seguridad para el **Cliente ECI**.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del flujo TS.
filterNr: uchar	Número del filtro de Fichero en el **Anfitrión ECI**.
dataFilter: dcrFilleFilterSpec *	Especificación del filtro para la extracción de datos.

Postcondiciones a la Petición:

- El filtro de esta sección será activado por el **Anfitrión ECI** hasta que se produzca una **reqDcrFileStop** o **reqDcrFileQuit** o que en una reqDcrFileFilter sea **dataFilter == NULL**.

9.6.2.4.5.2.4 Mensaje reqDcrFileAcqData

H→C reqDcrFileAcqData(ushort **mH**, uchar **filterNr**, uint **dataLen**, byte **data[]) →**
C→H resDcrFileAcqData (ushort **mH**, uchar **filterNr)**

- Este mensaje solicita al **Anfitrión ECI** que adquiera y envíe al **Cliente ECI** una o más secciones en el contexto del fichero o el flujo de medios identificado por **mH** y el filtro identificado por **filterNr**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del fichero en el que debe establecerse el filtro de sección por defecto.
filterNr: uchar	Número del filtro a programar. El valor está comprendido entre 0 y 7.
dataLen: uint	Número de bytes de **data**.
data[]: byte	Las secuencias de private_sections (bytes en el orden de la red) se definen en la sección 2.4.4.11 de [ISO/CEI 13818-1-1]. Si una sección tiene un error CRC no se transfiere al **Cliente ECI**.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del fichero o flujo de medios.
filterNr: uchar	Número del filtro que ha sido programado.

Los códigos de error conexos figuran en el Cuadro 9.6.2.4.5.2.4-1.

Cuadro 9.6.2.4.5.2.4-1 – Códigos de error de reqDerFileAcqData

Nombre	Descripción
ErrDcrAcqDataTimeout	Véase el Cuadro 9.6.2.4.7-1.
ErrDcrAcqDataDataErr	

9.6.2.4.6 API de palabra de control para la desaleatorización de ficheros

9.6.2.4.6.1 Generalidades

La sección de la API de desaleatorización de contenido permite que la clave esté disponible para la desaleatorización que realiza el **Cliente ECI**. El **Anfitrión ECI** debe, en primer lugar, poner a disposición una palabra de control transfiriendo al **Cliente ECI** el identificador de clave. Una vez que la clave está disponible, el **Anfitrión ECI** puede aplicar la palabra de control calculada al contenido (encriptado). Los mensajes de la API relacionados con la API de desaleatorización del contenido del Fichero **Asa de Medios** figuran en el Cuadro 9.6.2.4.6.1-1.

Cuadro 9.6.2.4.6.1-1 – API de desaleatorización del contenido del Fichero Asa de Medios

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqDcrFileKeyComp	A	H→C	0x20	Inicia cualquier cálculo necesario u otra actividad cualquiera del **Cliente ECI** para la disponibilidad de una palabra de control con identificador de clave.

9.6.2.4.6.2 Requisitos del procesamiento del Anfitrión ECI

9.6.2.4.6.2.1 Contenido con formato CENC ISOBMFF

En esta cláusula se definen los requisitos de procesamiento del **Anfitrión ECI** para la desaleatorización de contenido con formato CENC ISOBMFF+.

Es responsabilidad del **Anfitrión ECI** transferir oportunamente cualquier información sobre el identificador de claves (keyID) al **Cliente ECI** de forma que este pueda deducir/adquirir la palabra de control necesaria en el momento adecuado. Otras limitaciones que lo permitan deberían aplicarse al menos 30 segundos antes del uso previsto de la palabra de control.

La información sobre el identificador de clave (Key-ID) está incluida en varias cajas asociadas a las muestras de medios (secuencias de datos de medios (parcialmente) encriptados): véase por ejemplo la cláusula 5.4 de [b-DASH-IF V3]. Los datos en esas cajas permiten extraer los identificadores de clave, los IV y la identificación de datos sin encriptar y encriptados de las muestras de medios.

9.6.2.4.6.2.2 Contenido con formato DASH MPEG

Las especificaciones de la ECI no incluyen actualmente la información relativa a los formatos DASH MPEG que el **Anfitrión ECI** debe soportar.

9.6.2.4.6.3 Mensaje reqDcrFileKeyComp

H→C reqDcrFileKeyComp(ushort **mh**, byte **keyId**[MaxUuidLen]**) →**
C→H resDcrFileKeyComp(ushort **mH)**

- Este mensaje inicia el cálculo y cualquier otra actividad que necesite acometer el **Cliente ECI** para calcular una palabra de control identificada por KeyId y su puesta a disposición a fin de desencriptar el contenido.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del flujo TS.
keyId[MaxUuidLen]: byte	KeyID como UUID en el orden de los bytes de red.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del flujo TS.

Precondiciones a la Contestación:

1) La clave está disponible, o bien, se produce un error o vence el temporizador.

Información de la Semántica:

- El **Cliente ECI** informará de que se ha producido un error si no puede facilitarse oportunamente la palabra de control solicitada (60 segundos). Los **Clientes ECI** pueden seguir intentando adquirir la clave solicitada incluso después de que se haya informado de un error.
- Si se notifica un error, el **Anfitrión ECI** puede volver a enviar la **Petición**. Los **Anfitriones ECI** pueden enviar un máximo de 10 **Peticiones**.

Los códigos de error conexos figuran en el Cuadro 9.6.2.4.6.3-1.

Cuadro 9.6.2.4.6.3-1 – Códigos de error de reqDcrFileKeyComp

Nombre	Descripción
ErrDcrFileUserDelay	Véase el Cuadro 9.6.2.4.7-1.
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	
ErrDcrFileKeyIdUnknown	
ErrDcrFileKeyOverflow	

9.6.2.4.7 Códigos de error para la API de desencriptación del contenido de fichero y de flujo

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes de **Contestación** de esta API figuran en el Cuadro 9.6.2.4.7-1.

Todas las peticiones de **Asa de Medios** específicas de un fichero devuelven un código de error para el parámetro **Asa de Medios** si se aplican a un **Asa de Medios** no asociada a un fichero.

Cuadro 9.6.2.4.7-1 – Códigos de error de las API de sesión de medios para medios de ficheros y flujos

Nombre	Valor	Descripción
ErrDcrFileUserDelay	-256	Se ha producido una demora prolongada a la espera de información de entrada del **Usuario** necesaria para completar la operación. La operación no se ha completado.
ErrDcrFileCardMissing	-257	La **Tarjeta inteligente** necesaria para la sesión no está accesible/disponible.
ErrDcrFileServiceMissing	-258	No hay disponible un servicio externo al **CPE** (por ejemplo, servidor DRM) necesario para soportar las operaciones de desencriptación del **Cliente ECI**.
ErrDcrFileResourceMissing	-259	No hay disponible un servicio no definido interno al **CPE** necesario para el acceso al contenido o su desencriptación.
ErrDcrFileMmiMissing	-260	No está disponible el acceso del **Cliente ECI** al MMI.
ErrDcrFileDescrContinue	-261	El **Anfitrión ECI** continúa intentando desaleatorizar el contenido de este Fichero.

Cuadro 9.6.2.4.7-1 – Códigos de error de las API de sesión de medios para medios de ficheros y flujos

Nombre	Valor	Descripción
ErrDcrAcqDataTimeout	-262	Ha vencido una temporización asociada a la adquisición de datos.
ErrDcrAcqDataDataErr	-263	Se han obtenido secciones dentro del plazo del temporizador aunque con errores. Normalmente eso significa que el flujo está corrompido o que no cumple las especificaciones aplicables.
ErrDcrFileKeyIdUnknown	-300	keyId desconocido por el **Cliente ECI** /sistema de seguridad para este contenido.
ErrDcrFileKeyOverflow	-301	Se han producido demasiadas peticiones de identificadores de clave (Key-ID) en un breve periodo de tiempo; se esperan las **Contestaciones** del **Cliente ECI** a **Peticiones** previas de procesamiento.
ErrDcrFileKeyWithdrawn	-302	La clave ya no está disponible; el **Cliente ECI** ha retirado los derechos.

9.7 Conjunto de las API de acceso a los recursos de reencriptación del Anfitrión ECI

9.7.1 Introducción a las API de reencriptación

9.7.1.1 Lista de las API definidas en la cláusula 9.7

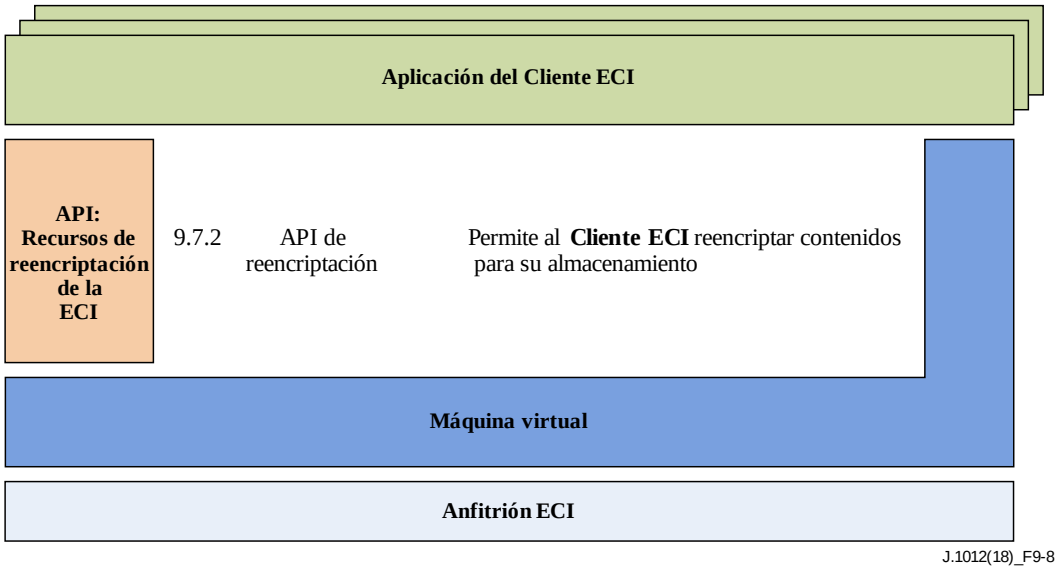


Figura 9.7.1-1 – Representación esquemática de las API definidas en la cláusula 9.7

En el Cuadro 9.7.1-1 figuran las API incluidas en la cláusula 9.7 y la Figura 9.7.1-1 ilustra la ubicación de las API definidas en esa cláusula con **arquitectura ECI**. Véase también [b-Menezes].

Cuadro 9.7.1-1 – Lista de las API definidas en la cláusula 9.7

Cláusula	Nombre de la API	Descripción
9.7.2.3	API de **conexión de exportación**	Permite al **Cliente ECI** establecer una **Conexión de Exportación** para un contenido importado.
9.7.2.5	API de **conexión de importación**	Permite al **Cliente ECI** importar un contenido que se distribuyó encriptado a través de la red de acceso y desencriptado bajo el control de un **Cliente ECI**.
9.7.2.6	API de desencriptación de **microcliente**	Permite al **Cliente ECI** desencriptar contenidos importados y reencriptados.

9.7.1.2 Concepto general de reencriptación

En el marco de la **ECI**, la reencriptación permite a un **sistema microDRM** independiente proteger el contenido distribuido por un **Cliente ECI** CA o DRM para aplicaciones adicionales internas o externas al CPE. El sistema de reencriptación en una implementación conforme con la **ECI** se denomina **Sistema microDRM**. Las aplicaciones de un **Sistema microDRM** pueden incluir, por ejemplo, el desfase temporal ("time-shifting"), el PVR y los flujos de datos. El **Cliente ECI** que realiza la reencriptación se denomina **Microservidor**. El cliente, sea o no conforme con la **ECI,** que pueden desencriptar el contenido se denomina **Microcliente**. La imagen del Cliente y las credenciales para la reencriptación pueden descargarse como un **Cliente ECI** ordinario, provisionado por un servidor maestro microDRM. En la Figura 9.7.1.2-1 se muestra una visión de conjunto del sistema (excluido el servidor maestro microDRM). En caso de almacenamiento local, el **Microservidor** y el **Microcliente** se implementan en un único dispositivo.

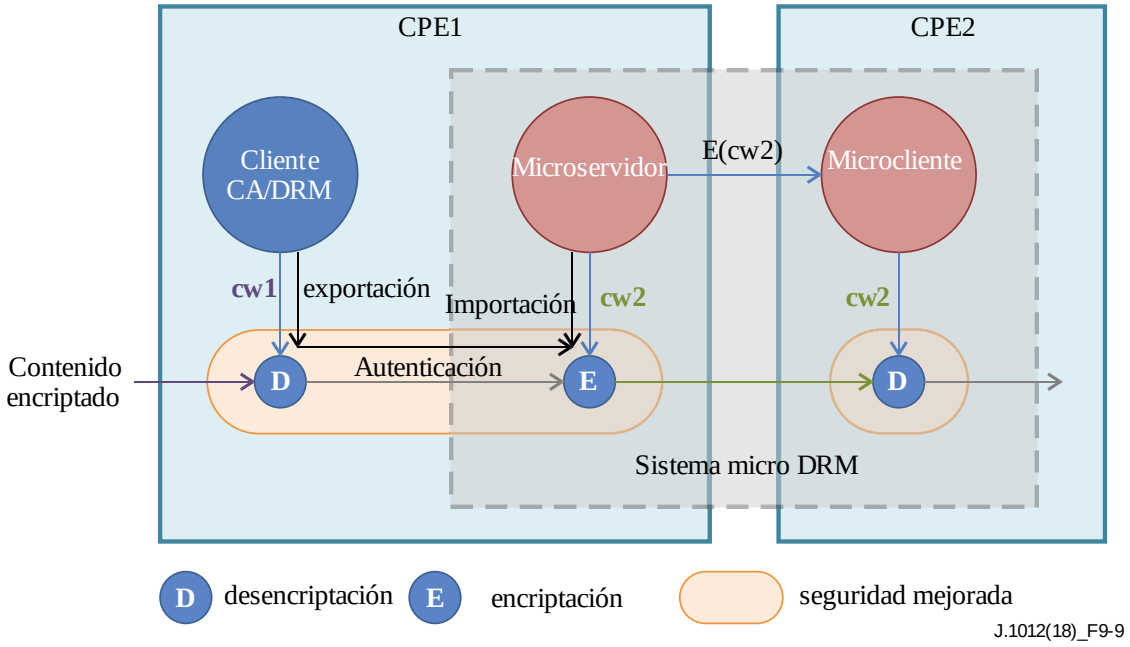


Figura 9.7.1.2-1 – Diagrama de un Sistema microDRM

El **Cliente ECI** CA/DRM que desencripta inicialmente el contenido puede controlar si está permitida la exportación del contenido a los **Sistemas microDRM** instalados. A tal fin, autentica al **Microservidor** mediante el sistema de **Seguridad Avanzada**; la autenticación permanece bajo el control del operador de CA/DRM. Una vez que se ha exportado el contenido, el **Sistema microDRM** tiene la responsabilidad de proteger el contenido. El sistema de **Seguridad Avanzada** soporta de forma segura la desencriptación, la reencriptación y la autenticación para la exportación. En la Figura 9.7.1.2-1 se ilustran los principios aplicables.

9.7.1.3 Visión general de la estructura de la API de reencriptación

En la Figura 9.7.1.3-1 se muestra un diagrama más detallado del papel de las diferentes API que participan en la reencriptación. El **Anfitrión ECI** proporciona al **Cliente ECI** que realiza la decodificación toda la información necesaria a través de la API de desencriptación. El **Cliente ECI** decodificador determina de forma segura la palabra de control para desencriptar el contenido mediante la API de Seguridad avanzada. Se autentican las propiedades del contenido fundamentales (marcas). La API de exportación permite al **Anfitrión ECI** solicitar al **Cliente ECI** decodificador el establecimiento de una conexión de exportación con el **Microservidor** deseado para la reencriptación. La API de Seguridad avanzada permite al **Cliente ECI** exportador autenticar al **Microservidor** importador. El **Anfitrión ECI** utiliza la API de importación para establecer la

conexión de exportación autorizada con un **Microservidor**. La API de reencriptación permite al **Anfitrión ECI** poner al **Microservidor** en un modo de operación correspondiente al formato de empaquetamiento de contenido y a la aplicación (flujos de datos, desfase temporal o almacenamiento), así como encriptar el contenido del **Microcliente** objetivo deseado (autenticado).

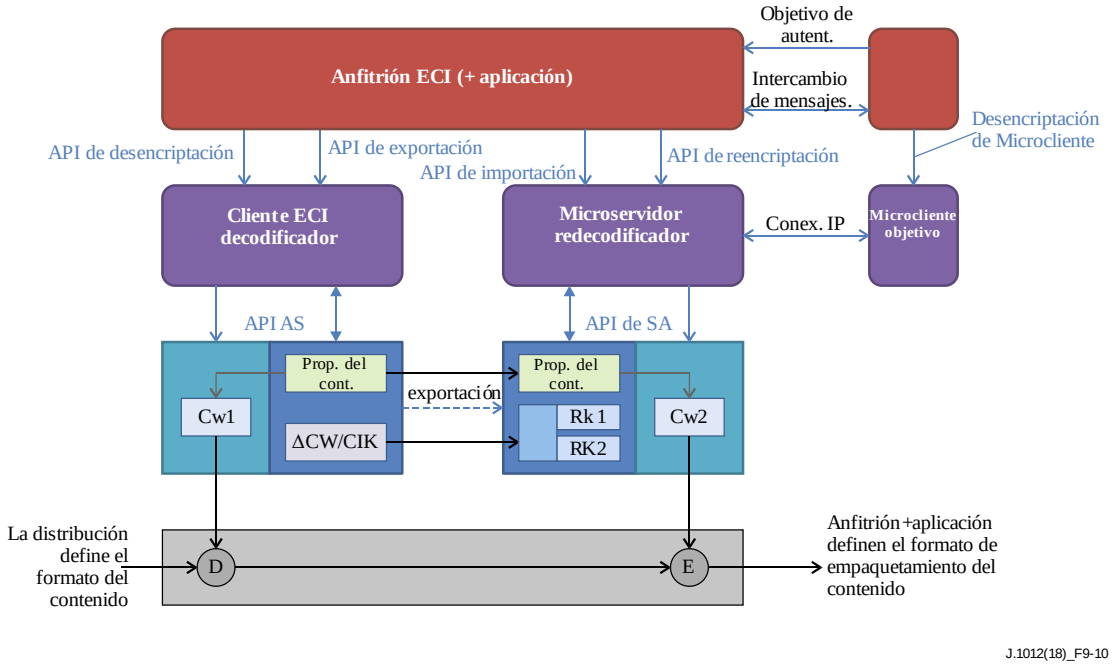


Figura 9.7.1.3-1 – Arquitectura de las funcionalidades de desencriptación y reencriptación

El esquema de la Figura 9.7.1.3-1 y la Figura 9.7.1.3-2 proporcionan una visión general de los principales mensajes de las API de desencriptación, control de exportación, control de importación, reencriptación y desencriptación del Microcliente. Muestra el contenido que fluye de izquierda a derecha: desde un primer **Cliente ECI** de prestación de CA/DRM sobre una **Conexión de exportación/importación** hasta un **Microservidor** que encripta el contenido previamente desencriptado que finalmente es decodificado por un **Microcliente objetivo**.

Las cuatro API anfitrión-cliente realizan las siguientes fases del procesamiento:

- La *fase de descubrimiento* permite a los **Clientes ECI** publicar sus opciones potenciales de interfuncionamiento al **Anfitrión ECI** (en colaboración con la aplicación). Ello permite al **Anfitrión ECI** establecer la posible correspondencia entre el contenido solicitado y un determinado **Cliente ECI**. Si el **Cliente ECI** elegido no posee los derechos adecuados para procesar el contenido, el **Anfitrión ECI** debe buscar otros **Clientes ECI**. En las redes del hogar y en aplicaciones de PVR distribuidos esto puede requerir protocolos de aplicación como DLNA, véase [b-DLNA]. La *fase de autenticación* permite al **Anfitrión ECI** establecer una conexión autenticada entre el **Cliente ECI** deseado y el **Microservidor** o entre el **Microservidor** y el **microCliente**. Las autenticaciones pueden ser implícitas, es decir, la prueba criptográfica de la autenticación puede estar integrada en la capacidad del **Cliente ECI** de desencriptar finalmente el contenido. La autenticación siempre sigue al flujo del contenido. En algunos casos es necesario un acuerdo inverso. Una conexión de importación puede tener que ser aprobada por el **Microservidor** con fines comerciales.

- La *fase de instanciación de sesión* permite al **Anfitrión ECI** reservar todos los recursos necesarios para desencriptar o encriptar contenidos en un determinado modo de funcionamiento asociado al **Asa de Medios**. Las conexiones de importación y **Objetivo** se definen para la reqEncrMhOpen en un **Microservidor**, o están implícitas en un **Cliente ECI** CA/DRM ordinario. Obsérvese que el **Anfitrión ECI** es responsable de atribuir cualquier recurso complementario, como los recursos para el procesamiento de la (des)aleatorización, la demultiplexación y la decodificación, en aras de conseguir un escenario completo de aplicación de medios. En última instancia, el **Cliente ECI** solicita la asignación de recursos de Seguridad avanzada (AS) y de desencriptación o encriptación utilizando la API de **Seguridad avanzada**.
- La *fase de control de sesión* permite al **Anfitrión ECI** iniciar y detener el procesamiento del contenido en **Asas de Medios**. Para un procesamiento sin discontinuidad del contenido en un trayecto, es necesario iniciar los **Clientes ECI** desde el destino hasta el origen: es decir, un **Cliente ECI** debe estar listo para procesar el contenido cuando este se presente.

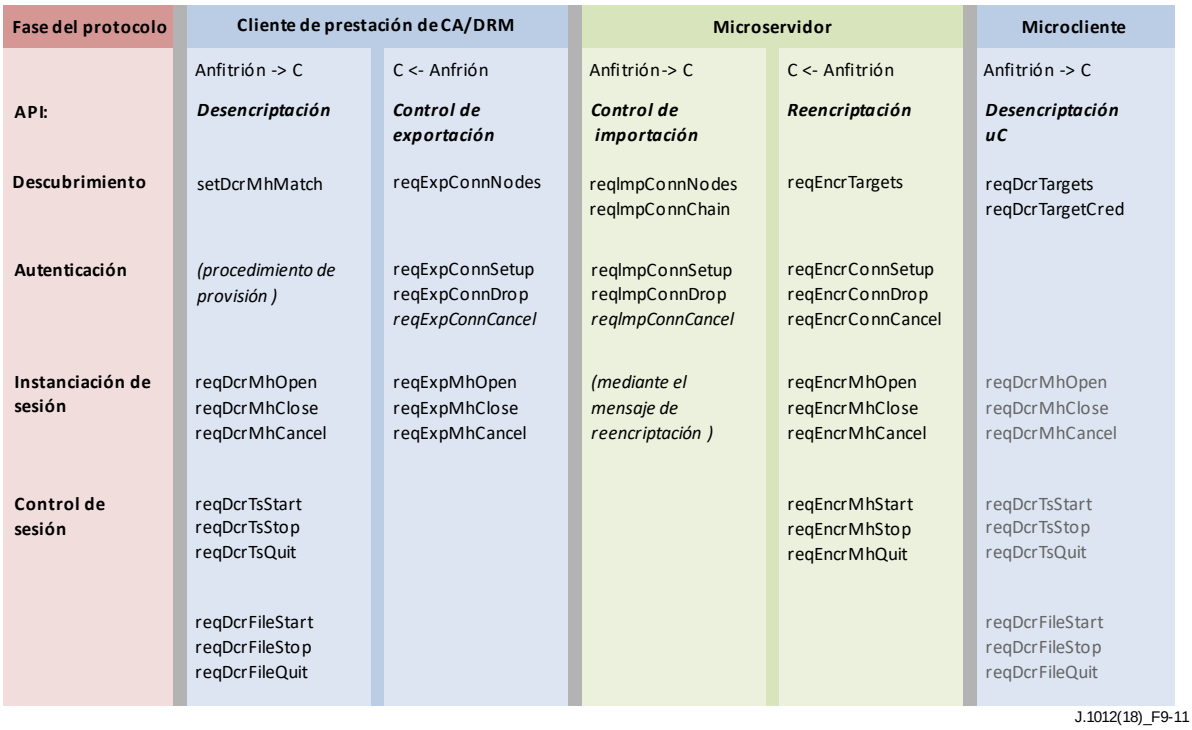


Figura 9.7.1.3-2 – Visión general de la API de encriptación/desencriptación y de importación/exportación

Los mensajes aplican una determinada sistemática en su denominación y semántica:

- La *fase de descubrimiento* permite al **Cliente ECI** publicar sus capacidades para la conexión con otro **Cliente ECI** o contenido. Los mensajes setDcrMhMatch, reqExpConnNodes, reqImpConnNodes, reqEncrTargets, reqDcrTargets solicitan al **Cliente ECI** su publicación (en forma de identidades).
- La *fase de autenticación* utiliza mensajes de establecimiento (*setup*), descarte (*drop*) y cancelación (*cancel*) para la creación de una conexión (autenticada), la desasignación de una conexión previa o la cancelación de la conexión por el **Cliente ECI**. La referencia para una conexión es una **Conexión de exportación** (**Cliente ECI** que exporta contenido), **Conexión de importación** (**Cliente ECI** que importa contenido) o una conexión **Objetivo** (**Microservidor** que encripta contenido para su ulterior desencriptación por un **Objetivo** y viceversa, por ejemplo, un **Microcliente** que desencripta contenido procedente de un **Microservidor**).

- La *fase de instanciación de sesión* utiliza la apertura (*open*), el cierre (*close*) y la cancelación (*cancel*) para crear y finalizar sesiones, todas ellas con relativas a un **Asa de Medios** de referencia común. Asimismo, la gestión de sesiones MMI y de los recursos de **Tarjeta inteligente** que necesita el **Cliente ECI** puede hacer referencia al **Asa de Medios** para permitir al **Anfitrión ECI** asociar una petición de diálogo de **Usuario** al contexto de su aplicación.
- La *fase de control de sesión* define distintos mensajes para la desencriptación de dos formatos de contenido específicos: formato de flujos de transporte y formato de ficheros. El procesamiento puede ser *iniciado* o *detenido* por el **Anfitrión ECI** y *abandonado* por el **Cliente ECI** por la falta de recursos o por alguna cuestión relativa a los derechos.

NOTA 1 – Para algunos sistemas de protección puede no ser necesario realizar un procesamiento de importancia en todas las fases. Sus **Clientes ECI** pueden realizar sólo procesos administrativos menores para algunos de los mensajes.

NOTA 2 – La naturaleza de los **Clientes ECI** en una **Conexión de importación/exportación** difiere de la relación entre un **Microservidor** y un **Microcliente**. En la **Conexión de importación/exportación** con **Clientes ECI** comparten el **Anfitrión ECI** y puede intercambiarse contenido mediante un mecanismo de exportación AS utilizando **Cadenas de certificados** de importación/exportación definidas en la **ECI**. El **Microservidor** y el **Microcliente** pueden utilizar un protocolo de su elección (característico del **Sistema microDRM**) para establecer la conexión en tanto que es compatible con el marco de la API y puede utilizar el **Sistema AS** a fin de establecer la autenticación y las claves comunes. El intercambio de contenido en una **Conexión de importación/exportación** es implícito (lo define el **Anfitrión ECI**); la autenticidad (con fines de exportación) del **Microservidor** será validada por el **Sistema AS**. El intercambio de contenido entre un **Microservidor** y un **Microcliente** requiere una sesión de **Asa de Medios** y control de sesión en el **Microservidor** y en el **Microcliente**.

9.7.2 API de Control de exportación ECI

9.7.2.1 Introducción

La **ECI** permite a los **Clientes ECI** exportar contenido decodificado al **Microservidor**, que asegurará su reencriptación al objeto de su redistribución (permitida) a otros dispositivos o su almacenamiento (permitido) para una reproducción posterior. A tal fin, la **ECI** define una estructura de **Certificado** con grupos de **Sistemas micoDRM** de exportación permitidos. Cada elemento de contenido decodificado está acompañado de la identificación del **Grupo de exportación** adecuado. A partir del **Grupo de exportación** debe existir una Cadena de Certificados que autorice la exportación al **Microservidor** seleccionado. El Sistema de seguridad avanzada procesa la cadena a fin de proporcionar un mecanismo de autorización de exportación robusto.

El **Cliente ECI** exportador es responsable de proporcionar **Certificados de grupo de exportación** y de todos los descendientes directos. El **Microcliente** importador es responsable de proporcionar la información de credencial complementaria que permita completar la cadena desde el **Cliente ECI** exportador al importador.

El **Anfitrión ECI** puede establecer una conexión de reencriptación desde un **Cliente ECI** desencriptador a un **Microservidor** encriptador. Una vez establecida la conexión, el **Anfitrión ECI** puede realizar la desencriptación y reencriptación del contenido utilizando sesiones del **Asa de Medios**. El **Sistema AS** garantizará una transferencia segura del contenido y la información de protección asociada desde el **Cliente ECI** decodificador al **Microcliente** basada en las credenciales proporcionadas a través del **Sistema AS**.

Los **Anfitriones ECI** apoyan a los **Clientes ECI en su** acceso a servicios de red a fin de recibir credenciales actualizadas de exportación e importación, por ejemplo, a través de la API del carrusel de datos (cláusula 9.5.4) y la API de HTTP IP (cláusula 9.4.4.6).

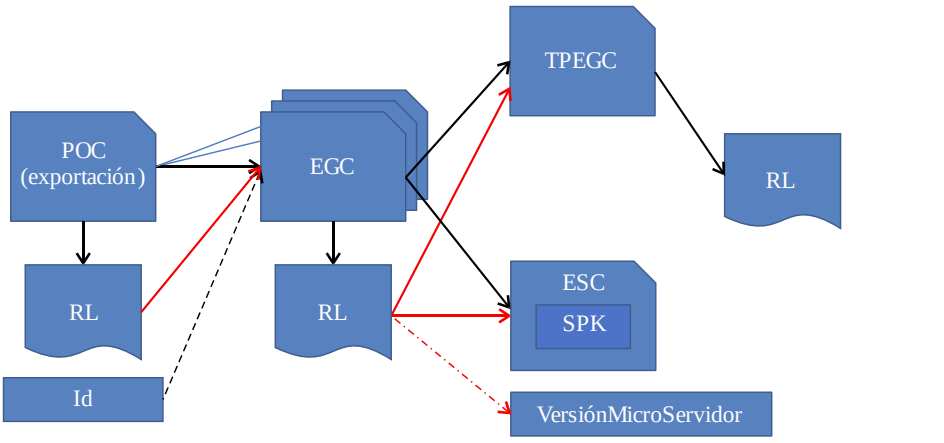
Con fines de reencriptación, el **Anfitrión ECI** y la aplicación deben establecer **Microclientes** autorizados que puedan decodificar el contenido. Puede tratarse de un **CPE** individual (con un cliente adecuado) o de un grupo (con una clave compartida). El **Anfitrión ECI** establece entonces una conexión autorizada entre el **Microservidor** y su correspondiente **Microcliente** (una para cada **Microcliente**). Para aplicaciones que utilicen el desfase temporal y la grabación puede almacenarse la información que necesita el **Cliente ECI** (junto con el contenido reencriptado) para una posterior decodificación del contenido. Para conexiones de flujos en tiempo real los mensajes de control de sesión que necesitan el **Microservidor** y el **Microcliente** pueden transferirse a través del **Anfitrión ECI** en caso de que los **Microclientes** y el **Microservidor** residan en el mismo dispositivo o bien puede establecerse una comunicación mediante una conexión IP directa entre los **Microclientes**.

NOTA – Los protocolos de comunicación y los aspectos de seguridad conexos para la comunicación entre **Clientes ECI** están fuera del alcance de la **ECI**.

9.7.2.2 Estructura de los certificados de exportación

9.7.2.2.1 Estructura general

El mecanismo de exportación de la **ECI** se basa en **Certificados**. La mayoría de los **Certificados** tienen una **Lista de Revocación** asociada que permite actualizar los permisos de exportación. En la Figura 9.7.2.2.1-1 se presenta la estructura del certificado para un control inmediato de la exportación de un **Cliente ECI** decodificador.



POC: Certificado de operación de plataforma
RL: Lista de Revocación
Id: ID de grupo de exportación (definido por los derechos sobre contenidos)
SPK: Clave pública del transmisor
EGC: Certificado de grupo de exportación
TPEGC: Certificado de grupo de exportación de un tercero
ESC: Certificado de sistema de exportación

Figura 9.7.2.2.1-1 – Estructura de distribución de certificados ECI

El certificado de la **Operación de Plataforma** (POC) del **Cliente ECI** es el **Padre** de los certificados del **Grupo de exportación**. El POC **ECI** tiene una lista de revocación especial para permitir al **Cliente ECI** controlar el certificado del **Grupo de exportación** y las versiones de las listas de revocación asociadas. Cada certificado de **Grupo de exportación** es **Padre** de certificados de exportación reales o de un **Grupo de exportación** adicional (descendiente). Existen dos tipos de certificados de exportación:

1) Un certificado de sistema de exportación (ESC) que identifica al **Microservidor** de exportación permitido mediante su **Clave pública de emisor**, lo que permite una autenticación inmediata. Además, el número de versión de la Lista de Revocación del ESC se utiliza al objeto de definir un número de versión mínimo para el **Microservidor**.

2) Un certificado de **Grupo de exportación** de un tercero (TPEGC) hace referencia a un **Certificado de grupo de exportación** gestionado por otra organización. Ello permite autenticar grupos heterogéneos más amplios de **Sistemas microDRM** con un único **Certificado** de exportación.

La estructura del **Certificado** de exportación de grupo de un tercero se ilustra en la Figura 9.7.2.2.1-2.

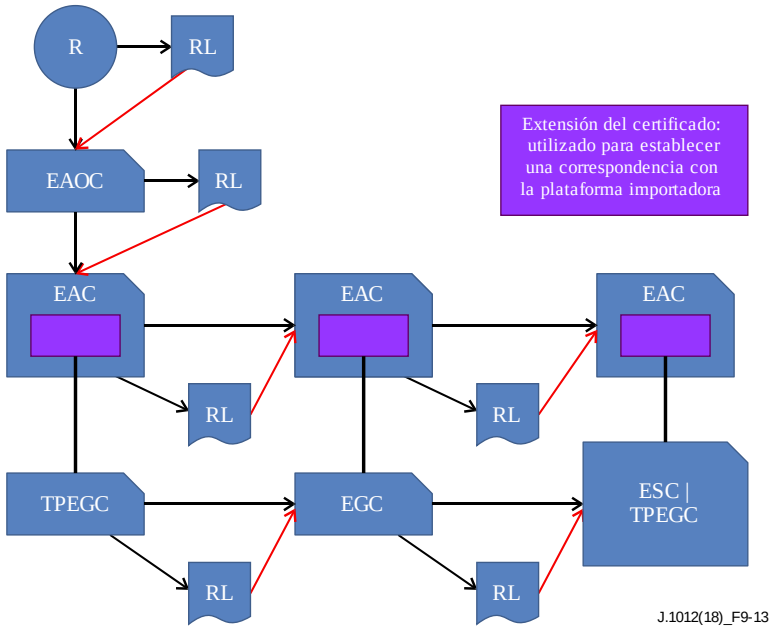


Figura 9.7.2.2.1-2 – Estructura del Certificado de exportación de grupo de un tercero

El **Certificado Raíz ECI** es el **Padre** de un Certificado de operador de autorización de exportación (EAOC). El **Certificado Raíz ECI** mantiene una Lista de Revocación especial para dichos **Certificados**. El EAOC es **Padre** de un Certificado de autorización de exportación (EAC). Se establece una correspondencia entre este certificado y un Certificado de **Grupo de exportación** de un tercero (TPEGC). Mediante este mecanismo se establece una doble autenticación de un grupo de un tercero para proporcionar seguridad adicional.

Un Certificado de **Grupo de exportación** de un tercero es **Padre** de algunos de los siguientes:

1) Un Certificado de **Grupo de exportación** (EGC), que puede ser **Padre** de otro EGC o de cualquiera de los certificados enumerados a continuación. Cada EGC tiene una **Lista de Revocación** asociada.
2) Un certificado del sistema de exportación (ESC).
3) Un Certificado (siguiente) del **Grupo de exportación** de un tercero (TPEGC).

Además, cada **Certificado** se verifica mediante un Certificado de autorización de exportación (EAC) concordante, de forma que se crea un árbol que se corresponde con el árbol TPEGC/EGC.

El Cuadro 9.7.2.2.1-1 ofrece una visión general de los **Certificados** y sus **Padres**.

Cuadro 9.7.2.2.1-1 – Resumen de los distintos certificados de exportación

Nombre del certificado	Abrev.	Descripción	Padre
Grupo de exportación (Export Group)	EGC	Este **Certificado** permite a **Clientes ECI** exportadores autenticar un conjunto (grupo) de **Microclientes** y/o a grupos autenticados de terceros a los que permiten la exportación. El **Grupo de exportación** aplicable se define como parte de un atributo relativo a derechos autenticado del contenido.	POC, TPEGC, EGC
Grupo de exportación de tercero (Third Party **Export Group**)	TPEGC	**Certificado** para autenticar un grupo de **Sistemas microDRM** gestionados por un tercero.	EGC, TPEGC
Operador de autorización de la exportación (Export Authorization Operator)	EAOC	**Certificado** que proporciona la base para que un **Operador** proporcione un servicio de autorización para **Grupos de exportación** de terceros. El **Certificado** es el **Padre** de árboles de **Certificados** de autorización de exportación para **Grupos de exportación** de terceros que este coautentica.	Raíz **ECI**
Autorización de exportación (Export Authorization)	EAC	Este **Certificado** proporciona la coautenticación de un Certificado de **Grupo de exportación** de terceros o de un Certificado de **Grupo de exportación** gestionado por un tercero.	EAC, EAOC
Sistema de exportación (Export System)	ESC	Este **Certificado** autentica el Certificado de **Operación de Plataforma** de un **Microcliente**.	EGC, TPEGC

9.7.2.2.2 Definiciones de Certificados de exportación

9.7.2.2.2.1 Certificado de Grupo de exportación y Lista de Revocación

La definición de **Certificados** para los **Certificados de Grupo de exportación ECI** (EGC) será conforme con la definición de ECI_certificate de la cláusula 5.2. El EGC utiliza el campo identificador de **Certificado**s **ECI** con los campos que se definen en el Cuadro 9.7.2.2.2.1-1.

Cuadro 9.7.2.2.2.1-1 – Definición del ID de Grupo de exportación ECI

Sintaxis	N.° de bits	Mnemónico
`ECI_EGC_Id {`		
`    type  /* véase el` Cuadro 5.3-1`*/`	4	uimsbf
`    export_group_id /* véase el` Cuadro 5.3-1 `*/`	20	uimsbf
`    export_group_version`	8	uimsbf
`}`		

Semántica:

Type	Valor conforme con el Cuadro 5.2-1.
export_group_id: entero	Id asignado al **Grupo de exportación** por la entidad que lo gestiona. Los valores 0x00000 y 0xFFFFF0-0xFFFFF están reservados.
export_group_version: entero	Versión del **Certificado del Grupo de exportación** con el identificador **export_group_id**.

Con fines de autenticación de **Certificado**s **Hijo**, el EGC estará acompañado de una Lista de Revocación con arreglo a la cláusula 5.3 y especialmente el Cuadro 5.3-1.

9.7.2.2.2.2 Certificado de Grupo de exportación de terceros y Lista de Revocación

La definición de **Certificados** para **Certificados de Grupo de exportación** de terceros de la **ECI** (TPEGC) será conformes con la definición general de ECI_certificate de la cláusula 5.2. El TPEGC utiliza el campo identificador de **Certificados ECI** con los campos que se definen en el Cuadro 9.7.2.2.2.2-1.

Cuadro 9.7.2.2.2.2-1 – Definición del campo identificador de TPEGC

Sintaxis	N.° de bits	Mnemónico
ECI_TPEGC_Id {		
type /* véase el Cuadro 5.2-1***/**	4	uimsbf
tp_export_group_id /* véase el Cuadro 5.3-1 ***/**	20	uimsbf
tp_export_group_version	8	uimsbf
}		

Semántica:

Type	Valor con arreglo al Cuadro 5.3-1.
tp_export_group_id: entero	Id asignado al **Grupo de exportación** de un tercero por la entidad que gestiona el **Grupo de exportación** de terceros. Los valores 0x00000 y 0xFFFFF0-0xFFFFF están reservados.
tp_export_group_version: entero	Versión del **Certificado del** g**rupo de exportación** con el identificador **tp_export_group_id**.

El campo extensión del TPEGC, tal como se define en el Cuadro 9.7.2.2.2.2-2, contendrá la estructura siguiente, utilizando las definiciones del **export_authorization_operator_id** que figuran en el Cuadro 9.7.2.2.2.4-1 y de **export_authorization_id** que figura en el Cuadro 9.7.2.2.2.5-1.

Cuadro 9.7.2.2.2.2-2 – Definición del campo extensión TPEGC

Sintaxis	N.° de bits	Mnemónico
ECI_ TPEGC _Extension {		
export_authorization_operator_id	20	uimsbf
export_authorization_id	20	uimsbf
padding(4)		
Extension_field **extension**		
}		

Semántica:

export_authorization_operator_id: entero	Identificador **ECI** del **Certificado de operador** de autorización de exportación que coautentica este **Certificado**.
export_authorization_id: entero	Identificador **ECI** del **Certificado** de autorización de exportación que coautentica este **Certificado** (véase la cláusula 9.7.1.2.2.5).
extension: Extension_field	Extensión de esta estructura.

Con fines de autenticación de **Certificados Hijo**, el TPEGC estará acompañado de una Lista de Revocación con arreglo a la cláusula 5.3 y el Cuadro 5.3-1.

9.7.2.2.2.3 Lista de Revocación raíz para Certificados de operador para autorización de la exportación

Con fines de autenticación, una cadena de autenticación para la exportación debe comenzar con una lista de revocación raíz de conformidad con la cláusula 5.3 y el Cuadro 5.3-1.

9.7.2.2.2.4 Certificado de operador para autorización de exportación

Las definiciones de **Certificados** para el **Certificado de Operador** de autorización de exportación **ECI** (TPEGC) serán conformes con la definición general de ECI_certificate de la cláusula 5.2. El EAOC utiliza el campo identificador de **Certificados ECI** con los campos que se definen en el Cuadro 9.7.2.2.2.4-1.

Cuadro 9.7.2.2.2.4-1 – Definición del campo identificador EAOC

Sintaxis	N.° de bits	Mnemónico
ECI_EAOC_Id {		
type **/* véase el** Cuadro 5.3-1***/**	4	uimsbf
export_authorization_operator_id **/* véase el** Cuadro 5.3-1 ***/**	20	uimsbf
export_authorization_operator_version	8	uimsbf
}		

Semántica:

type	Valor conforme con el Cuadro 5.3-1.
export_authorization_operator_id: entero	Id asignado al operador de autorización de exportación. Los valores 0x00000 y 0xFFFFF0-0xFFFFF están reservados.
export_authorization_operator_version: entero	Versión del **Certificado** de autorización de exportación con el identificador **export_authorization_operator_id**.

Con fines de autenticación de **Certificados Hijo**, el EAOC estará acompañado de una Lista de Revocación conforme con la cláusula 53 y el Cuadro 5.3-1.

9.7.2.2.2.5 Certificado de autorización de exportación y Lista de Revocación

Las definiciones de **Certificado** para **Certificados** de autorización de exportación **ECI** (EAC) serán conformes con la definición de ECI_certificate general de la cláusula 5.2, utilizando un campo extensión específico no vacío. El EAC utiliza el campo identificador de **Certificados ECI** con los campos que se definen en Cuadro 9.7.2.2.2.5-1.

Cuadro 9.7.2.2.2.5-1 – Definición del campo extensión EAC

Sintaxis	N.° de bits	Mnemónico
ECI_EAC_Id {		
type **/* véase el** Cuadro 5.3-1***/**	4	uimsbf
export_authorization_id **/* véase el** Cuadro 5.3-1 ***/**	20	uimsbf
export_authorization_version	8	uimsbf
}		

Semántica:

Type	Valor conforme con el Cuadro 5.3-1.
export_authorization_id: entero	Id asignado al **Certificado** de autorización de exportación (en el contexto de su **Padre**). Los valores 0x00000 y 0xFFFFF0-0xFFFFF están reservados.
export_authorization_version: entero	Versión del **Certificado** de operador de autorización de exportación cuyo identificador es **export_authorization_id**.

El campo extensión del EAC contendrá la estructura del **Certificado** que debe autorizarse para la exportación (véase la cláusula 5.1.3) excluyendo el campo **firma**, seguido de un campo extensión.

Con fines de autenticación de **Certificados Hijo**, los EAC estarán acompañados de una Lista de Revocación conforme con la cláusula 5.3 y el Cuadro 5.3-1, si estos deben autenticar Certificados **Hijo**.

9.7.2.2.2.6 Certificado del sistema de exportación

Las definiciones de **Certificado** para los **Certificados** del sistema de exportación de la **ECI** (ESC) serán conformes con la definición general de ECI_certificate de la cláusula 5.2. El campo public_key del **Certificado** contendrá el valor SPK utilizado por el **Microservidor**. El ESC utiliza el campo identificador de **Certificados ECI** con los campos que se definen en Cuadro 9.7.2.2.2.6-1.

Cuadro 9.7.2.2.2.6-1 – Definición del campo extensión ESC

Sintaxis	N.° de bits	Mnemónico
ECI_ESC_Id {		
type /* véase el Cuadro 5.3-1/	4	uimsbf
export_system_id /* véase el Cuadro 5.3-1 */	20	uimsbf
export_system_version	8	uimsbf
}		

Semántica:

Type	Valor conforme con el Cuadro 5.3-1.
export_system_id: entero	Id asignado al **Certificado** del sistema de exportación (en el contexto de su **Padre**). Los valores 0x00000 y 0xFFFFF0-0xFFFFF están reservados.
export_system_version: entero	Versión del **Certificado** del sistema de exportación cuyo identificador es **export_system_id**.

9.7.2.2.3 Validación de cadenas de certificados de exportación

El **Cliente ECI** exportador con una cadena previamente validada y con cadenas de autorización de exportación complementarias crearán la **Conexión de importación/exportación** solicitada. El **Cliente ECI** exportador y el **Microservidor ECI** importador, responsables de sus respectivas partes de las cadenas, proporcionarán al **Usuario** información en caso de que se produzcan problemas y/o intentos de adquisición de cadenas renovadas. El **Cliente ECI** proporcionará dichas cadenas de procesamiento al **Sistema AS** a fin de crear la **Conexión de exportación/importación** deseada. Si el **Sistema AS** encuentra errores de validación en alguna cadena o en la autorización de exportación complementaria, el **Cliente ECI** no podrá establecer la conexión requerida.

Los Certificados de autorización de exportación se utilizan para coautenticar un **Certificado** de exportación. Las reglas de procesamiento de la coautenticación son las siguientes:

1) El **Certificado** de autorización de exportación y el **Certificado** que debe ser coautenticado tienen firmas válidas (según definen sus respectivos **Padres**) y no han sido revocados.

2) Todos los datos del **Certificado** que debe ser coautenticado, excepto su firma, se comparan con los datos del correspondiente campo extensión del **Certificado** de autorización de exportación. Si no puede establecerse una correspondencia la coautenticación se considera fallida.

Para la creación de una **Conexión de exportación** el CPS seguirá las siguientes reglas de procesamiento:

1) Serán de aplicación todas las reglas de procesamiento del CPS para **Cadenas de certificados** enumeradas en la cláusula 5.4.2.

2) El CPS verificará si los tipos de **Hijos** de un **Certificado Padre** son adecuados de conformidad con el Cuadro 5.2-2.

3) El **Padre** de la **Cadena de exportación** del **Cliente ECI** exportador será el POC **ECI** del cliente. La Lista de Revocación adjunta para **Grupos de exportación** se aplicará a la validación de **Certificados de Grupo de exportación de Hijos**. El número de versión de la Lista de Revocación del POC para **Grupos de exportación** será mayor que la minClientVersion (véase [UIT-T J.1014]) del cliente.

4) El CPS aceptará como máximo 2 niveles de EGC para el **Cliente ECI** exportador. Es decir, un **Hijo** de un EGC de segundo nivel será un TPEGC o ESC.

5) El CPS asegurará que cualquier TPGC esté acompañado de un EAC coautenticado mediante una cadena (con Listas de revocación adjuntas) desde la raíz al EAOC y hasta el EAC. La versión de la Lista de Revocación Raíz para el Certificado de Operador de autorización de exportación se utilizará para determinar el número más elevado de versión de la Lista de Revocación para la "validación de la integridad del sistema".

6) El CPS garantizará que cualquier EGC, ESC y TPEGC que descienda de un TPEGC es coautenticado mediante un EAC que sea **Hijo** del EAC que validó el **Padre** de ese **Certificado**.

Los **Clientes ECI** exportadores y los **servidores microDRM** deben realizar un procesado previo basado de sus cadenas y proporcionar las versiones más recientes para evitar revocaciones en el CPS.

9.7.2.2.4 Protocolos de transporte para credenciales de exportación

9.7.2.2.4.1 Generalidades

Los **Clientes ECI** exportadores y los **Microservidores** pueden definir sus propios formatos para transportar datos de credenciales. La ECI define un formato de fichero normalizado para el transporte de esos datos. Los **Clientes ECI** pueden acceder a dichos ficheros normalizados a través de la API de acceso al carrusel **ECI** para medios de difusión. A tal fin, y para el aprovisionamiento en línea de Clientes, la **ECI** define llamadas estándar a la API web.

9.7.2.2.4.2 Formato de ficheros del árbol de exportación

El formato de fichero para el árbol de **Grupos de exportación** se define en el Cuadro 9.7.2.2.4.2-1.

Cuadro 9.7.2.2.4.2-1 – Definición del fichero del árbol de exportación ECI

Sintaxis	N.° de bits	Mnemónico
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**		
}		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para verificar el formato de los datos que siguen. Toma el valor de los tres caracteres ASCII de 8 bits 'EET'. Los **Clientes ECI** comprobarán el valor de este campo para verificar si un fichero **ECI** tiene el formato esperado para la integridad de los datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida; los demás valores están reservados. Los **Clientes ECI** ignorarán cualquier imagen con un número de versión no reconocido.
operator_id: ECI_Operator_Id	ID de **Operador** del **Cliente ECI** del árbol de exportación contenido en el fichero. El campo operator_version corresponde a la raíz de export_group_tree (árbol del grupo de exportación).
Platform-operation_id: ECI_Platform_Operation_Id	ID de la **Operación de Plataforma** del **Cliente ECI** del árbol de exportación incluido en el fichero.
export_group_tree: ECI_RL_Tree	Estructura ECI_RL_Tree, que comienza con la Lista de Revocación del **Grupo de exportación** para los **Grupos de exportación** y terminada en. En el caso de Certificados que no requieran una **Lista de Revocación** complementaria, esta estructura incluirá una **Lista de Revocación** vacía cuya firma no es preciso que concuerde con el **Certificado**.
extensions: Campo Extensión	Datos adicionales definidos por el Operador.

9.7.2.2.4.3 Formato de los Ficheros de cadenas de importación

El formato de los Ficheros de **Cadenas de importación** de un **Microservidor** se define en el Cuadro 9.7.2.2.4.3-1.

Cuadro 9.7.2.2.4.3-1 – Definición de los ficheros de cadenas de importación ECI

Sintaxis	N.° de bits	Mnemónico
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**		
}		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para verificar el formato de los datos que siguen. Toma valor de los tres caracteres ASCII de 8 bits 'EIC'. Los **Clientes ECI** comprobarán el valor de este campo para verificar si un fichero **ECI** tiene el formato esperado para la integridad de los datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida; los demás valores están reservados. Los **Clientes ECI** ignorarán cualquier imagen cuyo número de versión no sea reconocible.
operator_id: ECI_Operator_Id	ID del **Operador** del **Microservidor** al que está destinada esta **Cadena de importación**.
platform_operation_id: ECI_Platform_Operation_Id	ID de la **Operación de Plataforma** del **Microservidor** al que está destinada esta **cadena de importación**.
nr_chains: entero	Número de **Cadenas de importación** en este fichero.
eaoc_id: ECI_Operator_Id	ID del **Operador** de autorización de la **Cadena de importación**.
eac_id: ECI_Platform_Id	ID del EAC que coautoriza la **Operación de Plataforma** de la **Cadena de importación**.
import_chain: ECI_Certificate_Chain	**Cadena de Certificado ECI**, que remite el **Certificado de Operación de Plataforma** de importación al ESG que identifica al **Microcliente**. La cadena puede contener varios TPEGC. Cada **Cadena de importación** válida se representará por separado: es decir, si la cadena1 consta de dos subcadenas de terceros y la segunda subcadena también puede utilizarse por separado como **Cadena de importación**, se representará por separado. Para **Certificados** que no requieran una **Lista de Revocación** complementaria, esta estructura contendrá una **Lista de Revocación** vacía cuya firma no es preciso que concuerde con el **Certificado**.
extensions: Extension_field	Datos adicionales definidos por el Operador.

9.7.2.2.4.4 Formato de los Ficheros de autorización de exportación

El formato del fichero de autorización de **Cadenas de exportación** de un **Microservidor** se define en el Cuadro 9.7.2.2.4.4-1.

Cuadro 9.7.2.2.4.4-1 – Definición de los ficheros de autorización de exportación ECI

Sintaxis	N.° de bits	Mnemónico
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**		
}		
}		

Semántica:

magic: byte[3]	Número mágico utilizado para verificar el formato de los datos que siguen. Toma el valor de los tres caracteres ASCII de 8 bits 'EEA'. Los **Clientes ECI** comprobarán el valor de este campo para verificar si un fichero **ECI** tiene el formato esperado para la integridad de los datos adicionales.
image_header_version: byte	Versión del formato del encabezamiento de la imagen. El valor 0x01 corresponde a la versión actualmente definida; los demás valores están reservados. Los **Clientes ECI** ignorarán cualquier imagen con un número de versión no reconocible.
operator_id: ECI_Operator_Id	ID del **Operador** del **Microservidor** al que está destinado esta **Cadena de importación**.
Platform_operation_id: ECI_Platform_Operation_Id	ID de la **Operación de Plataforma** del **Microservidor** al que está destinado esta **Cadena de importación**.
nr_chains: entero	Número de cadenas de autorización de exportación en este fichero.
direct_flag: bit	Si el valor es 0b1 la siguiente cadena autoriza directamente una subcadena ESC y tanto **o_id** como **po_id** no son pertinentes. Si el valor es 0b0 la siguiente cadena autoriza la subcadena TPEGC y los valores de **o_id** y **po_id** representan los identificadores del **Certificado** de autorización.
o_id: ECI_Operator_Id	ID del **Operador** de un tercero, una **Cadena de exportación** de un tercero autenticada por la siguiente cadena de autenticación de exportación.
po_id: ECI_Platform_Öperation_Id	ID de la **Operación de Plataforma** de un tercero, una **Cadena de exportación** de tercero que es autenticada por la siguiente cadena de autenticación de exportación
chain: ECI_Certificate_Chain	**Cadena de certificado ECI** desde el **Certificado Raíz ECI** hasta el EAC que autentica el primer TPEGC, ESG.
extensions: Extension_field	Datos adicionales definidos por el Operador.

9.7.2.2.4.5 Carruseles de difusión que transportan credenciales de exportación

Los operadores pueden desplegar carruseles **ECI** tal como se define en la cláusula 7.7.2 para transportar las credenciales de exportación y/o importación de los **Clientes ECI** que decidan soportar. No obstante, para un **Cliente ECI** específico el **Anfitrión ECI** sólo tendrá que supervisar las actualizaciones de los DSI con una sola ubicación de un carrusel de datos. Es decir, para el transporte de credenciales de exportación o importación utilizando el formato de carrusel estándar, un **Operador** utilizará el mismo carrusel que transporta la Imagen de Cliente, las credenciales de la **Operación de Plataforma,** los datos de revocación, etc. para dicho **Cliente EC**I. Véase también la cláusula 7.7.2.1.

Los formatos de los datos de los módulos del carrusel serán los indicados en el Cuadro 7.7.2.6-1. Los módulos designados mediante un compatibilityDescriptor cuyo campo descriptorType sea igual a 0xB0 transportarán módulos con una única estructura ECI_Export_Tree_File, aquellos cuyo campo descriptorType sea 0xB transportarán módulos con una única estructura ECI_Import_Chain_File y los que tengan un campo descriptorType igual a 0xB2 transportarán módulos con una única estructura ECI_Export_Authentication_File.

Es recomendable que la supervisión que realiza el **Cliente ECI** de las actualizaciones en el carrusel coincida con las que realiza el **Anfitrión ECI** para datos del otro **Cliente ECI** en aras de una gestión energética eficiente.

9.7.2.2.4.6 Aprovisionamiento en línea de credenciales de exportación

La presente Recomendación reserva las estructuras URL siguientes de la API web a fin de permitir una estructura estándar para que los **Clientes ECI** accedan a credenciales de exportación desde el servidor en línea de un Operador.

Con relación a la cláusula 7.7.3 para la definición de tail_extension y los convenios de notación:

```
tail_extension* ::=
         client_export |
         client_import |
         client_exp_auth .
```

La notación tail_extension* indica otras extensiones que pueden existir en futuras versiones de la presente Recomendación.

Para la importación/exportación se definen las peticiones de la API web siguientes:

```
client_export ::= 'client-export/' operator_id '/' platform_operation_id .
```

A esta se devolverá la última versión del fichero árbol de exportación con el formato ECI_Export_Tree_File para el **Cliente ECI** designado por **operator_id**, **platform_operation_id**.

```
client_import ::= 'client-import/' operator_id '/' platform_operation_id .
```

A esta se devolverá la última versión del fichero **Cadena de importación** con el formato ECI_Import_Chain_File para el cliente **Microservidor** identificado mediante **operator_id**, **platform_operation_id**.

```
client_exp_auth ::= 'client-exp_auth/' operator_id '/' platform_operation_id .
```

A esta se devolverá la última versión del fichero autenticación de exportación con el formato ECI_Export_Authentication_File para el cliente **Microservidor** identificado mediante **operator_id**, **platform_operation_id**.

9.7.2.3 API de conexión de exportación

9.7.2.3.1 Generalidades

Los **Clientes ECI** pueden proporcionar información de exportación al **Anfitrión ECI**. Ello permite al **Anfitrión ECI** emparejar el sistema exportador con **Cadenas de importación** concordantes de **Microservidores.** El **Anfitrión ECI** (y la aplicación) pueden definir que las conexiones actuales se establezcan para todas las opciones posibles. Puede intentar conectar el **Cliente ECI** exportador con el correspondiente **Cliente ECI** importador enviando al **Cliente ECI** exportador una petición de conexión con la **Cadena de importación** del **Cliente ECI** importador objetivo. El **Cliente ECI** exportador así como el **Anfitrión ECI** pueden solicitar la cancelación de la conexión o su reinicialización en caso de actualización de las credenciales de importación. Los mensajes de **Conexión de exportación** disponibles figuran en el Cuadro 9.7.2.3.1-1.

Cuadro 9.7.2.3.1-1 – Mensajes de la API de conexión de exportación

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqExpConnNodes	A	H→C	0x0	El **Anfitrión ECI** solicita al **Cliente ECI** nodos con opción de exportación.
reqExpConnSetup	A	H→C	0x1	El **Anfitrión ECI** solicita al **Cliente ECI** que inicialice una **Conexión de exportación** con un **Cliente ECI** importador basada en la **Cadena de importación**.
reqExpConnDrop	A	H→C	0x2	El **Anfitrión ECI** cancela cualquier conexión inicializada previamente de un **Cliente ECI** exportador con un **Cliente ECI** importador.
reqExpConnCancel	A	C→H	0x3	El **Cliente ECI** termina una **Conexión de exportación** inicializada con un **Cliente ECI** importador.
reqExpMhOpen	A	H→C	0x4	El **Anfitrión ECI** solicita al **Cliente ECI** que establezca una sesión de exportación basada en una **Conexión de exportación** previamente inicializada.
reqExpMhClose	A	H→C	0x5	El **Anfitrión ECI** cierra una sesión de exportación.
reqExpMhCancel	A	C→H	0x6	El **Cliente ECI** cancela una sesión de exportación.

9.7.2.3.2 Mensaje reqExpConnNodes

H→C reqExpConnNodes() →
C→H resExpConnNodes(ExpConnOption **conn Nodes [])**

- Este mensaje solicita al **Cliente ECI** que devuelva su lista de posibles **Conexiones de exportación**; el mensaje **Contestación** devuelve la lista. Los códigos de error conexos figuran en el Cuadro 9.7.2.3.2-2.

Definición de los parámetros de la Contestación:

connNodes: ExpConn Option[]	La lista proporciona las identidades **ECI** de un tercero o los **Clientes ECI** con los que el **Cliente ECI** puede conectarse para la exportación. Cada opción tiene una prioridad: cuanto más elevada sea la prioridad menor será la probabilidad de que una exportación no se complete satisfactoriamente. ExpConnNode se define en el Cuadro 9.7.2.3.2-1.

Cuadro 9.7.2.3.2-1 – Definición del tipo ExpConnNode

```
typedef struct ExpConnNode {
   uint  targetType;
   uint  operatorId;
   uint  targetId;
   uint  targetPriority;
} ExpConnNode;
```

Definición de campos:

targetType: uint	Tipo de objetivo: el valor 1 es EAC (de tercero), el valor 2 es POC (exportación directa). Los demás valores no están definidos.
operatorId: uint	Representa el ID del Certificado **ECI** de 20 bits del **operador** de la exportación objetivo: export_authorization_operator_id para el objetivo EAC y operator_id para el objetivo POC.
targetId: uint	Representa el ID del Certificado **ECI** de 20 bits de la exportación objetivo, siendo export_authorization_ id para el objetivo EAC y platform_operation_id para el objetivo POC.
targetPriority: uint	La prioridad para seleccionar una exportación en particular es la suma de dos partes: • Valor en múltiplos de 1 024 que representa una prioridad (comercial) específica para la exportación que debe conectarse con un determinado **Microservidor**. • Valor comprendido entre 0 y 1 023 que representa una fracción menos 1 de 1 024, de los casos de uso de contenido previstos que pueden exportarse con este **Sistema microDRM** exportador. Los **Anfitriones ECI** utilizarán esta información para seleccionar automáticamente el **Sistema microDRM** más adecuado (siempre que el sistema con la máxima prioridad cumpla los requisitos de la aplicación para la aplicación de microDRM) y/o presentar lo anterior como una preferencia dirigida al **Usuario** en caso de selección manual.

Cuadro 9.7.2.3.2-2 – Códigos de error de reqExpNodeInfo

Nombre	Descripción
ErrExpConnNwAccess	Véase el Cuadro 9.7.2.3.9-1.
ErrExpConnAuthProblem	
ErrExpUninitState	

9.7.2.3.3 Mensaje reqExpConnSetup

H→C reqExpConn Setup (CertChainSerial **Import**, CertChainSerial **Auth[]**, ushort **connId) →**
C→H resExpConn Setup ()

- Este mensaje solicita al **Cliente ECI** que inicialice (o reinicie) la **Conexión de exportación** con identificador **connId** con el **Cliente ECI** con cuyo identificador es **clientId** utilizando la **cadena de importación** identificada por **Import**, las cadenas de autenticación de exportación identificadas por **Auth** y la cadena de **Cliente ECI** identificada por **Objetivo**.

Definición de los parámetros de la Petición:

Import: CertChainSerial	**Cadena de importación** (del TPEGC de exportación a ESC).
Auth: CertChainSerial[]	Cadenas de autenticación de exportación desde la Raíz al EAC que autentica al primer TPEGC en una única subcadena de tercero. Las cadenas incluidas en **Auth** están se ordenan desde el TPEGC de conexión exportador al POC importador.
connId: ushort	ID de la **Conexión de exportación**, asignado por el **Anfitrión ECI**.

Tipo CertChainSerial y definición de tipos de matriz

CertChainSerial es la representación en el orden de la de red (esquema "big endian") de `ECI_Certificate_Chain` tal como se define en el Cuadro 5.4.1-1, con relleno hasta alcanzar un múltiplo de 32 bits.

CertChainSerial[] se define mediante la siguiente estructura de datos (cuasi-C):

```
typedef struct CertChainSerial {
   uint   numberElements;     /* número de elementos en la matriz de la
                                    cadena */
   uint   elementIndex[];     /* índice del inicio de cada elemento en el
                                    contenedor de datos chainElements */
   uint   chainElements[];    /* contenedor de datos con representaciones
                                    numberElements SertChainSerial de las
                              sucesivas cadenas de la matriz. */
     } CertChainSerial;
```

Los elementIndex y chainElements se representarán mediante matrices de datos en línea en la estructura de datos certChainSerialArray.

Información de la Semántica:

- Los **Anfitriones ECI** pueden generar una petición reqExpConnSetup en nombre de una conexión existente para informar al **Cliente ECI** exportador de (potencialmente) nuevas credenciales de importación del **Cliente ECI** importador. A menos que la conexión actual pueda descartarse de inmediato, se recomienda que los **Clientes ECI** exportadores retrasen la renovación de la conexión con el **Cliente ECI** importador hasta que deje de haber sesiones activas.

Los códigos de error conexos figuran en el Cuadro 9.7.2.3.3-1.

Cuadro 9.7.2.3.3-1 – Códigos de error de reqExpConnSetup

Nombre	Descripción
ErrExpConnNwAccess	Véase el Cuadro 9.7.2.3.9-1.
ErrExpConnAuthProblem	
ErrExpUninitState	
ErrExpInvalidChain	

9.7.2.3.4 Mensaje reqExpConnDrop

H→C reqExpConnDrop(ushort **connId) →**
C→H resExpConnDrop()

- Este mensaje solicita al **Cliente ECI** que descarte una **Conexión de exportación** con el cliente identificado por **connId**.

Definición de los parámetros de la Petición:

connId: ushort	ID de la **Conexión de exportación**.

Precondiciones a la Petición:

1) Se ha establecido previamente una **Conexión de exportación** (identificada por **connId**).

Postcondiciones a la Contestación:

1) Se ha cerrado la **Conexión de exportación** (si existía).

Los códigos de error conexos figuran en el Cuadro 9.7.2.3.4-1.

Cuadro 9.7.2.3.4-1 – Códigos de error de reqExpConnDrop

Nombre	Descripción
ErrExpConnNone	Véase el Cuadro 9.7.2.3.9-1.

9.7.2.3.5 Mensaje reqExpConnCancel

C→H reqExpConnCancel(ushort **connId) →**
H→C resExpConnCancel()

- Este mensaje informa al **Anfitrión ECI** que el **Cliente ECI** ha dado por terminada la **Conexión de exportación** identificada por **connId**.

Definición de los parámetros de la Petición:

connId: ushort	ID asignada a la conexión.

Precondiciones a la Petición:

1) Se ha establecido previamente una **Conexión de exportación** identificada por **connId**.

9.7.2.3.6 Mensaje reqExpMhOpen

H→C reqExpMhOpen(ushort **mhExp**, ushort **mhDcr**, ushort **connId) →**
C→H resExpMhOpen(ushort **mhExp)**

- Este mensaje solicita al **Cliente ECI** la creación de una sesión de exportación identificada por el **mh** del **Asa de Medios** sobre la **Conexión de exportación connId**.

Definición de los parámetros de la Petición:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.
mhDcr: ushort	**Asa de Medios** para la sesión de desencriptación a exportar.
connId: ushort	ID asignado a la **Conexión de exportación**.

Definición de los parámetros de la Contestación:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.

Precondiciones a la Petición:

1) Se había establecido previamente la **Conexión de exportación connId**.
2) Se había establecido previamente la sesión de desencriptación **mhDcr**.

Postcondiciones a la Petición:

1) Se establece una **Conexión de exportación** o bien se ha producido error.

Información de la Semántica:

- El **Cliente ECI** exportador puede suspender y reanudar la exportación en una sesión existente, por ejemplo, basada en la inclusión de un **Grupo de exportación** de la conexión.

Los códigos de error conexos figuran en el Cuadro 9.7.2.3.6-1.

Cuadro 9.7.2.3.6-1 – Códigos de error de reqExpMhOpen

Nombre	Descripción
ErrExpConnNone	Véase el Cuadro 9.7.2.3.9-1.
ErrExpDcrMhNone	

9.7.2.3.7 Mensaje reqExpMhClose

H→C reqExpMhClose(ushort **mhExp) →**
C→H resExpMhClose(ushort **mhExp)**

- Este mensaje solicita al **Cliente ECI** el cierre de una sesión de exportación identificada mediante el **mh** del **Asa de Medios** sobre la **Conexión de exportación connId**.

Definición de los parámetros de la Petición:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.

Definición de los parámetros de la Contestación:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.

Precondiciones a la Petición:

1) Se ha establecido previamente una sesión de exportación **mhExp** que aún no ha terminado.

Postcondiciones a la Petición:

1) Se ha detenido la sesión de exportación **mhExp**.

Los códigos de error conexos figuran en el Cuadro 9.7.2.3.7-1.

Cuadro 9.7.2.3.7-1 – Códigos de error de reqExpMhClose

Nombre	Descripción
ErrExpMhNone	Véase el Cuadro 9.7.2.3.9-1.

9.7.2.3.8 Mensaje reqExpMhCancel

C→H reqExpMhCancel(ushort **mhExp) →**
H→C resExpMhCancel(ushort **mhExp)**

- Este mensaje informa al **Anfitrión ECI** de que el **Cliente ECI** ha detenido la sesión de exportación **mhExp**.

Definición de los parámetros de la Petición:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.

Definición de los parámetros de la Contestación:

mhExp: ushort	**Asa de Medios** asignado por el **Anfitrión ECI** a la **Conexión de exportación**.

Precondiciones a la Petición:

1) Se había establecido previamente una sesión de exportación **mhExp**.

2) El **Cliente ECI** ha dado por terminada la sesión.

9.7.2.3.9 Códigos de error de la API de conexión de exportación

Los valores y significados de los errores específicos de la API que pueden devolver los mensajes **Contestación** de esta API figuran en el Cuadro 9.7.2.3.9-1.

Cuadro 9.7.2.3.9-1 – Códigos de error de la API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrExpConnNwAccess	-256	El acceso a la red que proporciona información sobre la información solicitada no es posible o es inesperadamente lento y no pudo completarse.
ErrErrConnAuthProblem	-257	Se han detectado incoherencias internas en los datos aprovisionados que impiden completar la petición.
ErrEcxpConnUninitState	-258	El **Cliente ECI** requiere en primer lugar el aprovisionamiento y/o otras funciones operativas a fin de poder responder a la petición.
ErrExpConnInvalidChain	-259	Se ha detectado que una cadena proporcionada al **Cliente ECI** era inválida y/o la imposibilidad de autenticarla utilizando las Cadenas de autenticación.
ErrExpConnNone	-260	La conexión no existía.
ErrExpMhNone	-261	El **Cliente ECI** no soporta la sesión de exportación indicada por el **Asa de Medios**.
ErrExpDcrMhNone	-262	El **Cliente ECI** no soporta la sesión de desencriptación indicada por el **Asa de Medios**.

9.7.2.4 API de conexión de importación

9.7.2.4.1 Generalidades

Los **Clientes ECI** pueden proporcionar sus **Cadenas de importación** al **Anfitrión ECI**. Ello permite al **Anfitrión ECI** conectar al **Cliente ECI** importador a opciones de exportación concordantes de **Microservidores**. El **Anfitrión ECI** y la aplicación pueden establecer la conexión (o conexiones) creadas a partir de las opciones de conexión disponibles. El **Anfitrión ECI** puede iniciar el establecimiento de una conexión entre **Cliente ECI** exportador e importador solicitando en primer lugar al Cliente importador permiso para su conexión al **Cliente ECI** exportador. El Cliente importador puede rechazar dicha conexión, por ejemplo, sobre la base de consideraciones comerciales de su operador. Si se establece una conexión, el **Cliente ECI** importador así como el **Anfitrión ECI** pueden solicitar la cancelación o la reinicialización de la conexión en caso de actualización de las credenciales de importación.

Las cadenas de entrada se identifican por su primer nodo, es decir, los id **ECI** del EAOC y el EAC para el TPEGC. A ello se hace referencia en el Cuadro 9.7.2.4.1-1 como *nodo de importación*.

Cuadro 9.7.2.4.1-1 – Mensajes de la API de conexión de importación

Mensaje	Tipo	Dir.	Etiqueta	Descripción
reqImpConnNodes	A	H→C	0x0	El **Anfitrión ECI** solicita al **Cliente ECI** importador que le proporcione sus nodos de importación.
reqImpConnChain	A	H→C	0x1	**El Anfitrión ECI** solicita al **Cliente ECI** importador que le proporcione la cadena de entrada para un nodo de importación específico.
reqImpConnChainRenew	A	C→H	0x2	El **Cliente ECI** solicita al **Anfitrión ECI** que reinicialice la conexión utilizando una **Cadena de importación** actualizada.
reqImpConnSetup	A	H→C	0x3	El **Anfitrión ECI** solicita al **Cliente ECI** importador que inicialice una **Conexión de importación** con un **Cliente ECI** exportador específico a través de un nodo de importación.
reqImpConnDrop	A	H→C	0x4	El **Anfitrión ECI** descarta la **Conexión de importación** con el **Cliente ECI** exportador especificado.
reqImpConnCancel	A	C→H	0x5	El **Cliente ECI** termina la **Conexión de importación** con el **Cliente ECI** exportador especificado.

9.7.2.4.2 Mensaje reqImpConnNodes

H→C reqImpConnNodes () →
C→H resImpConnNodes(ImpConnNode **nodes[])**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Cliente ECI** importador que le proporciones sus nodos de importación.

Definición de los parámetros de la Contestación:

nodes[]: ImpConnNode	Matriz de nodos de importación y número de terceros intermediarios. La estructura de ImpConnNodes se define en el Cuadro 9.7.2.4.2-1.

Cuadro 9.7.2.4.2-1 – Definición de tipos de ImpConnOption

```
typedef str uct ImpConnNode {
   uint  targetType;
   uint  operatorId;
   uint  targetId;
   uint  intermediaries
} ImpConnNode;
```

Definición de los campos:

targetType: uint	Tipo de objetivo: 1 es EAC (de tercero), 2 es POC (exportación directa). Otros valores no están definidos.
operatorId: uint	Representa el ID del Certificado **ECI** de 20 bits del **Operador** de la importación del **Objetivo**: export_authorization_operator_id para el objetivo EAC, u operator_id para el objetivo POC.
targetId: uint	Representa el ID del Certificado **ECI** de 20 bits de la importación del **Objetivo**: export_authorization_ id para el objetivo EAC, o platform_operation_id para el objetivo POC.
intermediaries: uint	Representa el número de terceros intermediarios desde el nodo de entrada al POC del **Cliente ECI** importador. Los **Anfitriones ECI** seleccionarán la **Cadena de importación** más corta de entre las alternativas de opciones de exportación con la misma targetPriority para el **Cliente ECI** exportador.

Los códigos de error conexos figuran en el Cuadro 9.7.2.4.2-2.

Cuadro 9.7.2.4.2-2 – Códigos de error de reqExpConnInfo

Nombre	Descripción
ErrImpConnNwAccess	Véase el Cuadro 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpUninitState	

9.7.2.4.3 Mensajes reqImpConnChain y reqImpConnChainRenew

H→C reqImpConnChain(ImpConnNode **node)** →
C→H resImpConnChain(CertChainSerial **Import**, CertChainSerial **Auth[])**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Cliente ECI** importador que proporcione una cadena de entrada para un nodo de importación específico.

C→H reqImpConnChainRenew(CertChainSerial **Import**, CertChainSerial**Auth[])** →
H→C resImpConnChainRenew()

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** que reinicialice la conexión utilizando una **Cadena de importación** actualizada.

Parámetros de petición para reqImpConnChain:

node: ImpConnNode	Nodo de importación para el que se devolverá la **Cadena de importación** al **Anfitrión ECI**.

Definición de los parámetros de la Petición para reqImpConnChainRenew y definición de parámetros de la Contestación para reqImpConnChain:

Import: CertChainSerial	**Cadena de importación** (del TPEGC de exportación al ESC).
Auth: CertChainSerial[]	Cadenas de autenticación de exportación desde la Raíz al EAC que autentica el primer TPEGC en una única subcadena de un tercero. Las cadenas de **Auth** están ordenadas desde el TPEFC exportador al POC importador.

Precondiciones a la petición reqImpConnChainRenew:

1) Se ha establecido previamente una **Conexión de importación** con un **Cliente ECI** utilizando un elemento de la cadena aprovisionada.

Información sobre la semántica de reqImpConnChainRenew:

- El **Anfitrión ECI** transferirá la información de la cadena actualizada a los **Clientes ECI** exportadores afectados.
- Se recomienda que los Operadores proporcionen cadenas actualizadas con antelación suficiente al descarte de la cadena anterior de forma que se asegure la provisión ininterrumpida del servicio.

Los códigos de error asociados a reqImpConnChain figuran en el Cuadro 9.7.2.4.3-1.

Cuadro 9.7.2.4.3-1 – Códigos de error de reqImpConnChain

Nombre	Descripción
ErrImpConnNwAccess	Véase el Cuadro 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	

Los códigos de error asociados a reqImpConnChainRenew figuran en el Cuadro 9.7.2.4.3-2.

Cuadro 9.7.2.4.3-2 – Códigos de error de reqImpConnChainRenew

Nombre	Descripción
ErrImpConnNoConn	Véase el Cuadro 9.7.2.4.7-1.

9.7.2.4.4 Mensaje reqImpConnSetup

H→C reqImpConnStart (ImpConnNode **node**, ushort **exportClientId**, ushort **connId) →**
C→H resImpConnStart()

- Este mensaje permite al **Anfitrión ECI** solicitar al **Cliente ECI** importador que establezca una **Conexión de importación** con un **Cliente ECI** exportador específico a través de un nodo de importación.

Parámetros de la Petición:

node: ImpConnNode	Nodo de importación a través del que se establece la conexión.
exportClientId: ushort	Identificación por el **Anfitrión ECI** del **Cliente ECI** exportador.
connId: ushort	**ID** asignado a la **Conexión de importación**.

Información de la Semántica:

- El **Cliente ECI** puede rechazar la **Conexión de importación** sobre la base de consideraciones comerciales de su operador.

Los códigos de error conexos figuran en el Cuadro 9.7.2.4.4-1.

Cuadro 9.7.2.4.4-1 – Códigos de error de reqExpConnStart

Nombre	Descripción
ErrImpConnNwAccess	Véase el Cuadro 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnRefuseComm	
ErrImpConnUnknError	

9.7.2.4.5 Mensaje reqImpConnDrop

H→C reqImpConnDrop (ushort **connId) →**
C→H resImpConnDrop()

- Este mensaje permite al **Anfitrión ECI** descartar la **Conexión de importación** con el **Cliente ECI** exportador especificado.

Parámetros de la Petición:

connId: ushort	Identificación por el **Anfitrión ECI** de la **Conexión de importación** a descartar.

Precondiciones a la Petición:

1) Se ha inicializado previamente una **Conexión de importación** (identificada por **connId**).

Postcondiciones a la Contestación:

1) Se cierra la **Conexión de exportación** (si existe).

Los códigos de error conexos figuran en el Cuadro 9.7.2.4.5-1.

Cuadro 9.7.2.4.5-1 – Códigos de error de reqExpConnInfo

Nombre	Descripción
ErrImpConnNwAccess	Véase el Cuadro 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnNoConn	

9.7.2.4.6 Mensaje reqImpConnCancel

C→H reqImpConnCancel (ushort **connId) →**
**H→C resImpConnCancel(**)

- Este mensaje permite al **Cliente ECI** terminar la **Conexión de importación** con el **Cliente ECI** exportador especificado.

Parámetros de la Petición:

connId: ushort	**Conexión de importación** (identificada por connId) previamente inicializada.

Precondiciones a la Petición:

1) Previamente se había establecido una **Conexión de importación** con el cliente cuyo ID de cliente **Anfitrión ECI** es **exportClientId** que ahora está cerrada.

9.7.2.4.7 Códigos de error para la API de Conexión de exportación

Los valores y significados de los errores específicos de la API que pueden ser devueltos en los mensajes **Contestación** para esta API figuran en el Cuadro 9.7.2.4.7-1.

Cuadro 9.7.2.4.7-1 – Códigos de error de la API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrImpConnNwAccess	-256	El acceso a la red que proporciona información sobre la información solicitada era inesperadamente lento.
ErrImpConnAuthProblem	-257	Se han detectado incoherencias internas en los datos aprovisionados que impiden completar la solicitud.
ErrImpUninitState	-258	El **Cliente ECI** requiere en primer lugar el aprovisionamiento y/o otras funciones asociadas al funcionamiento a fin de poder responder a esta petición.
ErrImpConnRefuseComm	-259	Se ha detectado que una cadena proporcionada al **Cliente ECI** era inválida y/o no era posible autenticarla utilizando las cadenas de autenticación.
ErrImpConnRefuseComm	-260	El **Cliente ECI** importador declina conectarse al **Cliente ECI** exportador por motivos comerciales.
ErrImpConnUnknError	-261	El **Cliente ECI** importador ha detectado un error desconocido.
ErrExpConnNone	-262	La conexión no existía.

9.7.2.5 API de reencriptación

9.7.2.5.1 Generalidades

La API de reencriptación permite a un **Microservidor** volver a encriptar el contenido de una **Conexión de importación** específica de un grupo de clientes para su ulterior decodificación por un **Microcliente**. La decodificación puede tener que realizarse casi instantáneamente (conexión de flujo) y puede que no se permita la reproducción en una sesión posterior o alternativamente el contenido reencriptado puede almacenarse y reproducirse con un desfase temporal con información de desencriptación asociada para el **Microcliente** decodificador y ser decodificada más tarde por el **Microcliente**.

La fase de descubrimiento permite a la aplicación establecer una correspondencia entre un **Microservidor** y un posible **Objetivo** (**Microcliente** o grupo de **Microclientes**), e intercambiar la información de autenticación necesaria del **Microcliente** al **Microservidor** a fin de permitir la autenticación del **Microcliente** y disponer de lo necesario para un intercambio confiable del contenido. El **Anfitrión ECI** puede seleccionar un modo de comunicación bidireccional (basado en IP o mediante la transferencia de mensajes a través del **Anfitrión ECI**) de forma que se permitan protocolos de autenticación más sofisticados entre **Microservidor** y **Microcliente**.

Sobre la base de una conexión de reencriptación con el **Objetivo** y una **Conexión de importación**, el **Anfitrión ECI** puede instanciar una sesión de **Asa de Medios** del modo (modo de reencriptación, sincronización y formato de datos) que desea la aplicación y que soporta el **Microservidor**.

Una vez establecida la conexión de reencriptación, el **Anfitrión ECI** puede instanciar una sesión de **Asa de Medios** con un **Microservidor** e iniciar la reencriptación del contenido desde una **Conexión de importación** establecida para el **Objetivo** (**Cliente ECI** o grupo de **Clientes ECI**). Pueden instanciarse varias reencriptaciones simultáneas del mismo contenido, utilizando cada una su propia sesión de **Asa de Medios**. Es responsabilidad del **Anfitrión ECI** asegurar que el contenido de la sesión del **Asa de Medios** de exportación tiene su origen en el **Asa de Medios** de reencriptación autenticado de la **Conexión de exportación**. Una conexión errónea no autorizada producirá un fallo de autenticación de exportación.

Las palabras de control de reencriptación se aplican al contenido desencriptado importado y los nuevos marcajes (URI o de otro tipo) se aplican al contenido reencriptado utilizando el sistema AS (Seguridad avanzada).

Existen tres *modos de encriptación* principales:

1) Modo flujo en línea: tanto el **Microservidor** como el **Microcliente** están activos simultáneamente. Intercambian mensajes directamente (a través de un canal IP) o en forma de mensajes explícitos a través de sus **Anfitriones ECI**.
2) Modo flujo fuera de línea: el **Microservidor** encripta el contenido "al vuelo" y regularmente envía nuevos datos necesarios para que el **Microcliente** los desencripte. El resultado puede demorarse (modo de desfase temporal) o almacenarse.
3) Modo de almacenamiento fuera de línea: el **Microservidor** encripta el contenido y finalmente produce los datos que necesita el **Microcliente** inicialmente para la decodificación del contenido.

En la Figura 9.7.2.5.1-1 se describen esquemáticamente los distintos modos de encriptación.

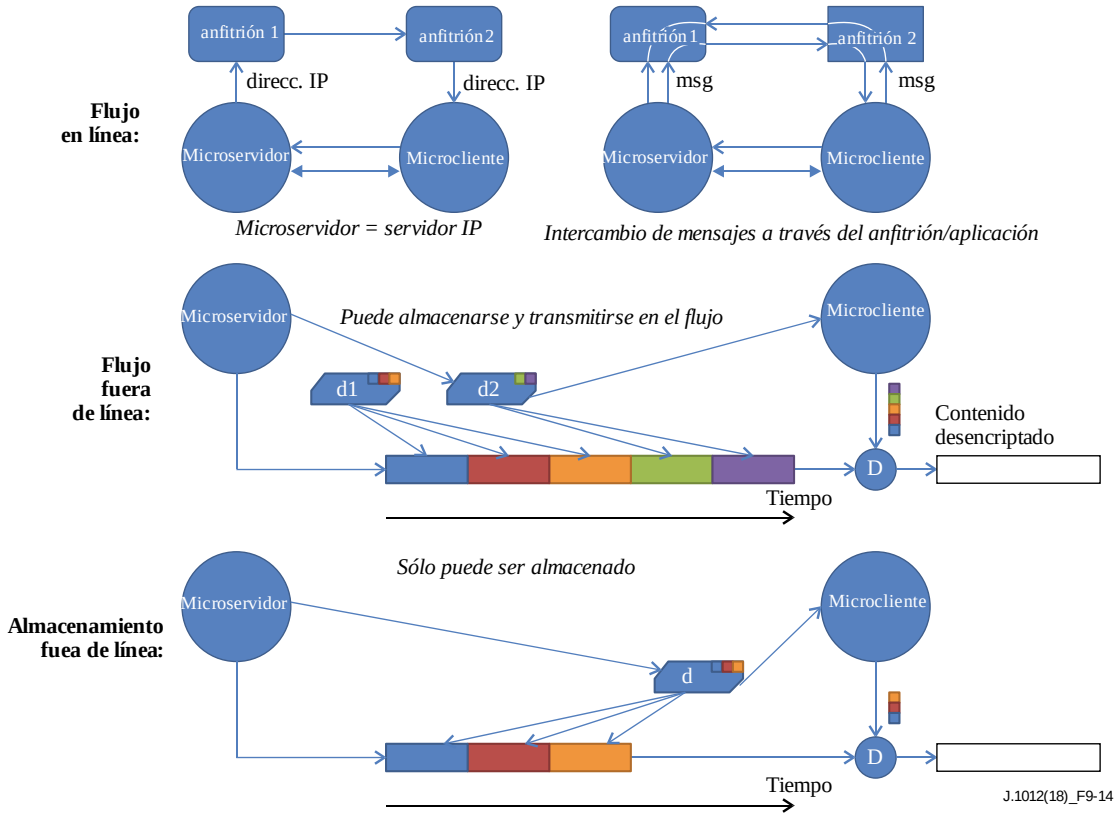


Figura 9.7.2.5.1-1 – Modos de encriptación para sesiones microDRM

Los datos necesarios para desencriptar el contenido a intercambiar en los dos modos de encriptación fuera de línea entre el **Microservidor** y el **Microcliente** pueden transferirse en los siguientes *modos formato de datos*:

1) Modo genérico: el **Microservidor** produce contenedores de datos opacos con la información necesaria para que el **Microcliente** desencripte el contenido.

2) Modo ISOBMFF (sólo para un *modo de sincronización* en modo de fichero): el **Microservidor** genera cajas PSSH para su inclusión en un fichero ISOBMFF [ISO/CEI 14496-12]. El **Anfitrión ECI** puede usarlas para crear ficheros ISOBMFF mediante la inclusión adecuada de las cajas PSSH en cajas MOOV o MOOF ISOBMFF.

El *modo sincronización* soporta dos mecanismos para la asociación de la palabra de control correcta a una sección de contenido, con aplicación a todos los modos de reencriptación arriba indicados:

1) En el modo flujo de transporte (alternancia de bits) el **Microservidor** produce secciones ECM que el **Anfitrión ECI** puede empaquetar e insertar en el flujo de transporte. El ECM se inserta antes del criptoperiodo para el que proporciona información que permita el cálculo de la palabra de control.

2) En el modo fichero el **Microservidor** produce palabras de control encriptadas a las que hacen referencia mediante identificadores de clave (KeyID) explícitos en la información de desencriptación suplementaria. El **Anfitrión ECI** debe preservar la asociación del KeyID de la sección de contenido encriptado con una palabra de control específica de forma que el **Microcliente** pueda producir la palabra de control correcta para la desaleatorización.

En el modo fuera de línea, la sincronización de los datos adicionales necesarios para la desencriptación o el cálculo del KeyId o los ECM hacen referencia explícita a la relación de dependencia temporal de los datos relativos al KeyId o al número de ECM.

No todos los **Microservidores** tienen que soportar todos los modos de operación. Durante la inicialización, e inmediatamente después del uso de la API de descubrimiento, un **Microservidor** señaliza los modos que soporta (combinación del modo encriptación, modo formato de datos y modo sincronización).

Una vez instanciada la sesión del **Asa de Medios**, esta puede ser iniciada y detenida por el **Anfitrión ECI** y ser cancelada por el **Cliente ECI**.

Los mensajes de la API de reencriptación figuran en el Cuadro 9.7.2.5.1-1.

Cuadro 9.7.2.5.1-1 – Mensajes de la API de reencriptación

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setEncrModes	set	C→H	0x0	El **Microservidor** informa al **Anfitrión ECI** de los modos que soporta (modo encriptación, modo formato de datos y modo sincronización).
reqEncrTargets	A	H→C	0x1	El **Anfitrión ECI** solicita al **Microservidor** que proporcione los nodos **Objetivo** que puede autenticar para la desencriptación.
reqEncrConnSetup	A	H→C	0x2	El **Anfitrión ECI** solicita al **Cliente ECI** la creación de una conexión **Objetivo** de reencriptación y la preautenticación del **Objetivo** de reencriptación para una ulterior referencia en el establecimiento de una sesión de **Asa de Medios**.
reqEncrConnDrop	A	H→C	0x3	El **Anfitrión ECI** solicita al **Cliente ECI** que descarte cualquier información sobre una conexión de reencriptacion anteriormente preautenticada.
reqEncrConnCancel	A	C→H	0x4	El **Cliente ECI** cancela una conexión **Objetivo** de encriptación previamente establecida.
reqEncrMhOpen	A	H→C	0x5	El **Anfitrión ECI** solicita al **Cliente ECI** que abra una sesión de **Asa de Medios** para reencriptar el contenido de una **Conexión de importación** entrante para una conexión de reencriptación establecida.
reqEncrMhClose	A	H→C	0x6	El **Anfitrión ECI** cierra la **Sesión de reencriptación** con el **Cliente ECI**.
reqEncrMhCancel	A	C→H	0x7	El **Cliente ECI** termina la **Conexión de importación** con el **Cliente ECI** exportador especificado.
reqEncrMhStart	A	H→C	0x8	El **Anfitrión ECI** solicita al **Cliente ECI** que inicie la operación de reencriptación para una sesión de **Asa de Medios**.
reqEncrMhStop	A	H→C	0x9	El **Anfitrión ECI** solicita al **Cliente ECI** que detenga una operación de reencriptación para una sesión de **Asa de Medios**.
reqEncrMhQuit	A	C→H	0xA	El **Cliente ECI** informa al **Anfitrión ECI** que ha terminado la operación de reencriptación del **Asa de Medios**.
reqEncrIpServer	A	H→C	0xB	El **Anfitrión ECI** solicita la dirección de servidor IP de un **Microservidor** para permitir que los **Microclientes** creen conexiones IP.
reqEncrMsgSend	A	C→H	0xC	El **Microservidor** solicita al **Anfitrión ECI** que envíe un mensaje al **Objetivo** de una sesión de **Asa de Medios.**
reqEncrMsgRecv	A	H→C	0xC	El **Anfitrión ECI** proporciona al **Microservidor** un mensaje procedente de un **Objetivo** de una sesión de **Asa de Medios**.
reqEncrTsData	A	C→H	0xE	El **Microservidor** proporciona al **Anfitrión ECI** datos que deben ser enviados al **Microcliente objetivo** de un **Asa de Medios** para su desencriptación, incluyendo información de sincronización relativa al ECM.
reqEncrTsEcm	A	C→H	0xF	El **Microservidor** envía una sección ECM que necesita el **Microcliente** para la desencriptación en el siguiente periodo de criptográfico.
reqEncrFileData	A	C→H	0x10	El **Microservidor** proporciona al **Anfitrión ECI** un mensaje que debe ser enviado al **Microcliente objetivo** de un **Asa de Medios** para su desencriptación, incluyendo información de sincronización conexa del KeyID.

9.7.2.5.2 Mensaje setEncrModes

C→H setEncrModes(EciEncrModes **modes)**

- Este mensaje permite al **Microservidor** informar al **Anfitrión ECI** sobre los modos que soporta (modo encriptación, modo formato de datos y modo sincronización).

Definición de los parámetros de la Petición:

modes: EciEncrModes	Modos de encriptación que soporta el **Microservidor**. El tipo EciEncrModes se especifica en el Cuadro 9.7.2.5.2-1.

Cuadro 9.7.2.5.2-1 – Definición de tipos de EciEncrModes

```
Typedef uint EciEncrModes;
```

Definición a nivel de bits:

Nombre	Bit	Modo del Microservidor soportado el valor 0b1
OnlineIpMode	0	Soporta el modo IP en línea.
OnlineMsgMode	1	Soporta el modo mensajes en línea.
OfflineStreamMode	2	Soporta el modo flujo fuera de línea.
OfflineStorageMode	3	Soporta el modo almacenamiento fuera de línea.
OfflineDataMode	4	Soporta contenedores de formato de datos por defecto para los datos a desencriptar en modo fuera de línea. No es pertinente si no se selecciona ningún modo fuera de línea.
OfflineIsobmffMode	5	Soporta las cajas PSSH con formato ISOBMFF para los datos a desencriptar en modo fuera de línea. No es pertinente si no se selecciona ningún modo fuera de línea.
SyncTs	6	Sincroniza las palabras de control con el formato del flujo de transporte, alternando en relación con el contenido periodos criptográficos delimitados por bits.
SyncFile	7	Sincroniza con formatos de tipo fichero utilizando la identificación de KeyID para asociar secciones de contenido a sus palabras de control.
Otros	RFU	Reservado para uso futuro.

9.7.2.5.3 Mensaje reqEncrTargets

H→C reqEncrTargets() →
C→H resEncrTargets(EncrTarget **target[])**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** que proporcione los objetivos de encriptación que puede autenticar.

Definición de los parámetros de la Contestación:

target: EncrTarget[]	Lista de objetivos de encriptación que el **Microservidor** puede autenticar. La definición de tipos de TargetClient (cliente objetivo) se especifica en el Cuadro 9.7.2.5.3-1.

Cuadro 9.7.2.5.3-1 – Definición de tipos de EncrTarget

```
typedef struct EncrTarget {
   uint  targetType;
   byte  target[8];
} EncrTarget;
```

Definición de campos:

targetType: uint	Tipo de objetivo de encriptación: el valor 1 significa cliente individual, el valor 2 significa grupo de clientes, los demás valores están reservados para uso futuro.
target: byte[8]	ID que representa el objetivo. El valor se define dentro del alcance del **Sistema microDRM**. La concordancia del **Anfitrión ECI** se establece en términos de igualdad de los campos **targetType** (**tipo de objetivo**) y **target** (**objetivo**).

Información de la Semántica:

- Puede existir concordancia entre el **Anfitrión ECI** y potenciales **Microclientes objetivo** basada en el **Objetivo**. La localización de potenciales **Microclientes** candidatos depende de la aplicación y/o el **Anfitrión ECI**.
- Los **Anfitriones ECI** que deseen realizar funciones de PVR local y con desfase temporal (utilizando un medio de almacenamiento integrado o conectado/en red donde puedan almacenar contenido encriptado y datos conexos) pueden intentar establecer una correspondencia con un **Microservidor** que opere en modo **OfflineStreamMode** y **Microclientes** instalado en el mismo **Anfitrión ECI**.

9.7.2.5.4 Mensaje reqEncrConnSetup

H→C reqEncrConnSetup(ushort **targetConnId**, EciEncrTarget **target**, ushort **credLen**, byte **cred[])**

C→H resEncrConnSetup(ushort **targetConnId)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** la creación de una conexión de reencriptación con el **Objetivo** y (pre)autenticar el **Objetivo**. Los códigos de error se definen en el Cuadro 9.7.2.5.19-1.

Definición de los parámetros de la Petición:

targetConnId: ushort	ID para referencia ulterior al **Objetivo** entre el **Anfitrión ECI** y el **Microservidor**.
target: EciEncrTarget	ID que representa el **Objetivo** a efectos de autenticación. El valor se define dentro del alcance del **Sistema microDRM**. La correspondencia que establece el **Anfitrión ECI** se define en función de la igualdad de los campos targetType (**tipo de objetivo)** y **target** (**objetivo**).
credLen: ushort	Longitud en bytes del parámetro **cred**.
cred: byte[]	Información de la credencial del **Objetivo** que debe autenticar el **Microservidor**.

Definición de los parámetros de la Contestación:

targetConnId: ushort	ID para referencia ulterior al **Objetivo** entre el **Anfitrión ECI** y el **Microservidor**.

Información de la Semántica:

- Si el **targetConnId** coincide con un **targetConnId** previamente utilizado por el **Anfitrión ECI,** pero que no ha sido descartado posteriormente, se sustituye o actualiza el **Objetivo** previo asociado con **targetConnId**.

Precondiciones a la Petición:

1) El **Objetivo** debe ser igual a un **Objetivo** previamente suministrado al **Anfitrión ECI** por el **Microservidor** en un mensaje **resEncrTargets**. Si no es así, se devuelve un error para este parámetro.
2) El **Objetivo** debe corresponder con un **Objetivo** proporcionado por el **Microcliente** y permitir la autenticación utilizando **cred**.

Postcondiciones a la Contestación:

1) Se devuelve el estado de autenticación. Obsérvese que el resultado no es necesariamente concluyente y podría proporcionar credenciales equivocadas dando lugar, por ejemplo, a un contenido encriptado que no puede ser decodificado.

2) El **Anfitrión ECI** puede hacer referencia al **Objetivo** (pre)autenticado mediante **targetConnId**.

Cuadro 9.7.2.5.4-1 – Códigos de error de reqEncrConnSetup

Nombre	Descripción
ErrEncrAuthFail	Véase el Cuadro 9.7.2.5.19-1.
ErrEncrAuthInconclusive	

9.7.2.5.5 Mensaje reqEncrConnDrop

H→C reqEncrConnDrop(ushort **targetConnId) →**
C→H resEncrConnDrop(ushort **targetConnId)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** que descarte información de una conexión de reencriptación anteriormente preautenticada.

Definición de los parámetros de la Petición:

targetConnId: ushort	Id de la conexión **Objetivo** que el **Microservidor** debe eliminar.

Definición de los parámetros de la Contestación:

targetConnId: ushort	Id de la conexión **Objetivo** eliminada del **Microservidor**.

Precondiciones a la Petición:

1) El **Microservidor** debe tener un **targetConnId**.

Precondiciones a la Contestación:

1) El **Microservidor** no asocia en lo sucesivo un **targetConnId** con una conexión **Objetivo** preautenticada y ha liberado cualquier recurso asociado con la preautenticación de **targetConnId**.

9.7.2.5.6 Mensaje reqEncrConnCancel

C→H reqEncrConnCancel(ushort **targetConnId) →**
H→C resEncrConnDrop(ushort **targetConnId)**

- Este mensaje permite al **Microservidor** informar al **Anfitrión ECI** que ha cancelado una conexión de reencriptación anteriormente preautenticada.

Definición de los parámetros de la Petición:

targetConnId: ushort	Id de la conexión **Objetivo** que ha sido cancelada por el **Microservidor**.

Definición de los parámetros de la Contestación:

targetConnId: ushort	Id de la conexión **Objetivo** que ha sido cancelada por el **Microservidor**.

Precondiciones a la Petición:

1) El Microservidor debe tener un **targetConnId**.

Precondiciones a la Contestación:

1) El valor de **targetConnId** ha sido desasignado y puede ser reasignado por el **Anfitrión ECI** como parte de un mensaje ulterior reqEcnrConnSetup.

9.7.2.5.7 Mensaje reqEncrMhOpen

H→C reqEncrMhOpen(ushort **mh**, ushort **impConn**, ushort **targetConnId**, EncrMode **mode) → C→H resEncrMhOpen(**ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Cliente ECI** la apertura de una sesión de **Asa de Medios** para reencriptar contenido bajo el control del **Microservidor** desde una **Conexión de importación** a fin de enviarlo a un objetivo preautenticado. Los códigos de error se definen en el Cuadro 9.7.2.5.7-1.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación que debe abrirse, asignado por el **Anfitrión ECI.**
impConn: ushort	Id de la conexión de entrada cuyo contenido debe ser reencriptado.
targetConnId: ushort	Id de la conexión **Objetivo** cuyo contenido debe ser reencriptado.
mode: EncrMode	Especificación del modo único (modo encriptación, modo formato de datos y modo sincronización) de funcionamiento del **Microservidor**, seleccionado de entre las capacidades de los nodos del **Microservidor** indicas mediante **setEncrModes**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que debe abrirse, asignado por el **Anfitrión ECI**.

Precondiciones a la Petición:

1) El **Anfitrión ECI** ha reservado todos los recursos necesarios para la sesión que va a crearse.
2) El **Anfitrión ECI** establece **impConn** y **targetConnId** con el **Microservidor**.

Precondiciones a la Contestación:

1) Si el resultado es satisfactorio el **Microservidor** ha reservado todos los recursos que normalmente requiere la reencriptación de contenido para la sesión solicitada. Ello debe incluir el acceso a cualquier recurso externo (servidores DRM, **tarjetas inteligentes**, etc.) normalmente requerido para una operación de desencriptación.

   NOTA – Se excluyen recursos requeridos con carácter excepcional o que normalmente puedan obtenerse cuando se necesitan.

2) Si se devuelve **ErrEncrUserDelay**, el **Microservidor** queda pendiente de los datos de entrada del **Usuario** para la apertura de la sesión (por ejemplo, para acceder a una **Tarjeta inteligente** o la autenticación de un **Usuario**). El **Anfitrión ECI** puede repetir el envío de la **Petición reqEncrMhOpen** (con los mismos parámetros) hasta que se devuelva un resultado positivo, un error definitivo o alternativamente pueda enviarse **reqEncrMhClose** para dar por terminada la sesión pendiente. El **Microservidor** puede realizar la cancelación mediante **reqEncrMhCancel** si no puede obtener los datos de entrada de **Usuario** que son necesarios.

Cuadro 9.7.2.5.7-1 – Códigos de error de reqEncrMhOpen

Nombre	Descripción
ErrEncrUserMissing	Véase el Cuadro 9.7.2.5.19-1.
ErrEncrCardMissing	
EnrEncrServiceMissing	
ErrEncrResourceMissing	
ErrEncrMmiMissing	
ErrEncrClientAuthError	

9.7.2.5.8 Mensaje reqEncrMhClose

H→C reqEncrMhClose(ushort **mh) →**
C→H resEncrMhClose(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** cerrar una **Sesión de reencriptación** con el **Microservidor**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación que debe cerrarse.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que debe cerrarse.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** se encuentra en un estado abierto (o bien se producirá un error).

Precondiciones a la Contestación:

1) Se liberan los recursos que el **Microservidor** necesita para mantener la sesión.
2) El Cliente cierra el estado de **mh**.

9.7.2.5.9 Mensaje reqEncrMhCancel

C→H reqEncrMhCancel(ushort **mh**, uchar **reason) →**
H→C resEncrMhCancel(ushort **mh)**

- Este mensaje permite al **Cliente ECI** cerrar una **Sesión de reencriptación** con el **Cliente ECI** exportador especificado (**Microservidor**).

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación cancelada por el **Microservidor**.
reason: uchar	Motivos para cancelar la sesión de desencriptación. Los valores se definen en el Cuadro 9.7.2.5.9-1.

Cuadro 9.7.2.5.9-1 – Valores de los motivos de reqEncrMhCancel

Nombre	Valor	Descripción
EncrMhUndefined	0x00	Se ha producido en el **Microservidor** un error no definido que requiere que cancele la sesión.
EncrMhCardMissing	0x01	La **Tarjeta inteligente** es necesaria para la reencriptación, pero no pudo ser (re)conectada satisfactoriamente ni colaborar en la reencriptación del contenido en un tiempo razonable.
EncrMhServiceMissing	0x02	No se dispone de un servicio (externo al **CPE**) que soporte la prestación por el **Microservidor** en un tiempo razonable de servicios de encriptación necesarios para mantener una sesión de desencriptación.
EncrMhResourceMissing	0x03	El **Microservidor** no dispone de un recurso (externo al **CPE**) necesario para prestar servicios de reencriptación en un tiempo razonable (sin incluir **DcrMhMmiMissing**).
EncrMhMmiMissing	0x04	El **Microservidor** no ha podido obtener en un tiempo razonable un recurso de sesión MMI para la interacción del **Usuario** necesario para mantener la **Sesión de reencriptación**.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que se ha cancelado.

Precondiciones a la Petición:

1) El **Cliente ECI** ha liberado los recursos que necesita específicamente para la sesión.

Postcondiciones a la Petición:

1) El **Anfitrión ECI** puede liberar cualquier recurso relacionado con el **Asa de Medios**.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** ha cerrado la sesión del **Asa de Medios**.

9.7.2.5.10 Mensaje reqEncrMhStart

H→C reqEncrMhStart(ushort **mh) →**
C→H resEncrMhStart(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** que inicie una operación de reencriptación para una sesión de **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación que debe iniciarse.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que ha sido iniciada.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** se encuentra en estado abierto (o bien se producirá un error).

Precondiciones a la Contestación:

1) La sesión del **Asa de Medios** se ha iniciado (o bien se ha producido un error).

Información de la Semántica:

- La encriptación del contenido se realizará conforme el **Cliente ECI** exportador proporcione el contenido.
- Los conflictos relativos a la URI o los errores del **Cliente ECI** exportador para autenticar el **Microservidor** para la exportación de contenido no producirán contenidos encriptados, el estado de la URI de control de salida del **Microservidor** se fija con OcAnyOthers igual a 0b1, poniéndose a 0b0 los restantes bits de control de salida (lo que significa que no se permite ninguna salida). El **Microservidor** continuará intentando reencriptar contenido cómo y cuándo esté permitido.
- Los mensajes de inicialización del **Microcliente** se proporcionan en los mensajes establecidos a tal fin. Para sesiones con el modo de reencriptación **OfflineStreamMode,** los primeros datos de inicialización para desencriptar el contenido se producen en un breve lapso tras el mensaje **resEncrMhStart**.
- En el envío de una segunda **reqEncrMhStart** antes de finalizar el proceso de encriptación dará por terminado el proceso anterior e iniciará el siguiente.

9.7.2.5.11 Mensaje reqEncrMhStop

H→C reqEncrMhStop(ushort **mh) →**
C→H resEncrMhStop(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** que detenga la operación de reencriptación para una sesión de **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación a finalizar.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que se ha finalizado.

Precondiciones a la Petición:

1) El estado de la sesión del **Asa de Medios** es iniciada (o se producirá un error).

Precondiciones a la Contestación:

1) La sesión del **Asa de Medios** se ha finalizado.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI** puede reutilizar el valor de la sesión del **Asa de Medios**.

Información de la Semántica:

- En sesiones con modo de encriptación **OfflineStorageMode** los datos finales de la desencriptación se producen antes de que el **Microservidor** envíe **resEncrMhStop**. Esto también es aplicable para cualquier dato final de desencriptación que sea necesario para la desencriptación en otros tipos de sesiones.

9.7.2.5.12 Mensaje reqEncrMhQuit

C→H reqEncrMhQuit(ushort **mh**, uchar **reason) →**
C→H resEncrMhQuit(ushort **mh)**

- Este mensaje permite al **Microservidor** informar al **Anfitrión ECI** que ha finalizado la operación de reencriptación asociada al **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación que se ha finalizado.
reason: uchar	Motivos según se indica en el Cuadro 9.7.2.5.9-1.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que se ha finalizado.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** estaba en el estado de iniciada pero que ahora es finalizada.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** conoce que el estado de encriptación de la sesión es no iniciada.

Información de la Semántica:

- En caso de errores de naturaleza casi permanente el **Microservidor** también puede cancelar la sesión del **Asa de Medios**.
- Si el **Microservidor** puede producir datos de desencriptación válidos antes de finalizar la **Sesión de reencriptación**, los datos finales de desencriptación en sesiones con modo de encriptación **OfflineStorageMode** se producen antes de que el **Microservidor** envíe **resEncrMhQuit**. Esto también es válido para cualquier dato de desencriptación final que pueda ser necesario para la desencriptación en otros tipos de sesiones.

9.7.2.5.13 Mensaje reqEncrIpServer

H→C reqEncrIpServer(ushort **mh) →**
C→H resEncrIpServer(ushort **mh**, Addrinfo **addr)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microservidor** la dirección IP **Objetivo** para las conexiones IP entrantes procedentes de **Microclientes**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación que necesita una dirección IP para mensajes o conexiones entrantes.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación que necesita una dirección IP para los mensajes o conexiones entrantes.
addr: Addrinfo	Protocolo/dirección/puerto IP para los mensajes o conexiones entrantes de un **Microcliente**.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta en modo **OnlineIpMode**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** conoce que el estado de encriptación de sesión no iniciada.

Información de la Semántica:

- El intercambio IP entre **Microcliente** y **Microservidor** es específico del Sistema microDRM. Incluye la elección del protocolo y de cualquier convenio para la terminación de una conexión o el intercambio en una sesión de flujo de contenido.
- Este mensaje puede generarse en una sesión de **Asa de Medios** en la que el proceso de reencriptación aún no haya comenzado.

Cuadro 9.7.2.5.13-1 – Códigos de error de reqEncrIpServer

Nombre	Descripción
ErrEncrIpNone	Véase el Cuadro 9.7.2.5.19-1.

9.7.2.5.14 Mensaje reqEncrMsgSend

C→H reqEncrMsgSend(ushort **mh**, uint **length**, byte **msg[]) →**
C→H resEncrMsgSend(ushort **mh)**

- Este mensaje permite al **Microservidor** solicitar al **Anfitrión ECI** el envío de un mensaje al **Microcliente** o **Microclientes Objetivo** (en caso de que el objetivo sea un grupo) asociados al **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación en la que debe enviarse un mensaje al **Microcliente Objetivo**.
length: uint	Longitud en bytes del campo **msg**.
msg[]: byte	Mensaje a enviar al **Microcliente**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** se abre en modo **OnlineMsgMode**.

Precondiciones a la Contestación:

1) El mensaje se ha enviado al **Microcliente**: el **Anfitrión ECI** está listo para aceptar una nueva **reqEncrMsgSend**.

Información de la Semántica:

- El **Anfitrión ECI** podrá procesar y enviar al menos un mensaje al **Microcliente** en cualquier momento. Los mensajes deben entregarse en orden. El **Anfitrión ECI** no está obligado a proporcionar memoria intermedia específica con capacidad simultánea para más de una petición **reqEncrMsgSend** pendiente. Una implementación segura del **Microservidor** debería utilizar **resEncrMsgSend** como mecanismo de toma de contacto del flujo de control.
- La fiabilidad del mecanismo de transmisión del **Anfitrión ECI** deberá ser suficiente para que no se produzcan errores en aplicaciones ordinarias (pérdida de mensajes o alteración del orden en uno de cada 10 000). Se recomienda que en el caso de aplicaciones en las que pueda perderse de forma permanente información de acceso esencial para contenido encriptado y/o durante las cuales pueda degradarse la calidad de la visualización, se tomen medidas preventivas adicionales al nivel de la aplicación.

9.7.2.5.15 Mensaje reqEncrMsgRecv

H→C reqEncrMsgRecv(ushort **mh**, uint **length**, byte **msg[]) →**
C→H resEncrMsgRecv(ushort **mh)**

- Este mensaje permite que el **Anfitrión ECI** proporcione al **Microservidor** un mensaje del **Microcliente Objetivo**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación en la que el **Microservidor** recibe un mensaje del **Microcliente Objetivo**.
length: uint	Longitud en bytes del campo **msg**.
msg: byte[]	Mensaje que debe recibir el **Microservidor**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** de la sesión de encriptación para la que se necesita una dirección IP para mensajes o conexiones entrantes.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta en modo **OnlineMsgMode**.

Precondiciones a la Contestación:

1) El **Microservidor** ha procesado el mensaje y está listo para aceptar una nueva **reqEncrMsgRecv**.

Información de la Semántica:

- El **Microservidor** procesará al menos un mensaje en cualquier momento. El **Microservidor** no está obligado a proporcionar memoria intermedia específica con capacidad simultánea para más de una petición **reqEncrMsgSend** pendiente, aunque debería vigilar que está en disposición de procesar un mensaje posterior manteniendo sus demás necesidades en términos de capacidad de respuesta. Una implementación segura del **Anfitrión ECI** debería utilizar **resEncrMsgRecv** como mecanismo de toma de contacto del flujo de control.

- La fiabilidad del servicio de envío entre **Microcliente** y **Microservidor** es como el definido para **reqEncrMsgSend** en la cláusula 9.7.2.5.14.

9.7.2.5.16 Mensaje reqEncrTsData

C→H reqEncrTsData(ushort **mh**, TsSync **sync**, uint **length**, byte **msg[]) →**
C→H resEncrTsData(ushort **mh)**

- Este mensaje permite al **Microservidor** proporcionar al **Anfitrión ECI** datos a enviar al **Microcliente Objetivo** de un **Asa de Medios** para permitir la desencriptación del contenido, incluida información de sincronización relacionada con el ECM.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación.
sync: TsSync	Sincronización de esta información relativa a un ecmId asociado al contenido. La información al respecto figura en el Cuadro 9.7.2.5.16-1.
length: uint	Longitud en bytes del mensaje a enviar.
msg: byte[]	Mensaje a enviar al **Microcliente**.

Cuadro 9.7.2.5.16-1 – Definición de typedef de TsSync

```
typedef struct TsSync {
   uint  ecmId;
   uint  precTime;
} TsSync;
```

Definición de campos:

ecmId: uint	Número de identificación de un ECM asociado con el contenido al que debe preceder este mensaje de datos para el **Microcliente**.
precTime: uint	Tiempo real expresado en unidades de 100 ms, con un máximo de 300 segundos, de precedencia de este mensaje respecto a la reproducción del contenido antes de la aplicación de un ECM con ecmId al proceso de decodificación de contenido.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación para la que se necesita una dirección IP para mensajes o conexiones entrantes.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta, la sesión está en *modo de reencriptación* **OfflineStream** u **OfflineStorage**, utiliza el *modo formato de datos* **OfflineDataMode** y el *modo sincronización* **SyncTs**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** está listo para recibir el próximo mensaje de datos.

Información de la Semántica:

- El **Anfitrión ECI** debe garantizar que se facilitan al **Microcliente** los datos en línea con los requisitos de sincronización junto con el contenido encriptado.
- El **Anfitrión ECI** almacenará adecuadamente en una memoria intermedia los datos del mensaje (como datos asociados al contenido) y debe responder al siguiente mensaje dentro del plazo de tiempo propuesto en [b-UIT-T J Supl. 7].
- El **Microservidor** puede producir uno o más mensajes de datos antes del inicio de una **Sesión de reencriptación** cuando está en modo **OfflineStream**.

- El **Microservidor** generará como máximo un mensaje de datos al final de la sesión de encriptación en modo **OfflineStorage**. Este mensaje de datos puede estar precedido por el ECM con el que supuestamente debe sincronizarse. Por lo tanto, se trata del modo "almacenamiento fuera de línea". Normalmente, el **Microcliente** debe procesar este mensaje de datos antes que cualquier contenido y mensajes ECM.

9.7.2.5.17 Mensaje reqEncrTsEcm

C→H reqEncrTsEcm(ushort **mh**, uint **ecmId**, uint **length**, byte **ecm[]) →**
C→H resEncrTsEcm(ushort **mh)**

- Este mensaje permite al **Microservidor** generar una sección ECM necesaria para la desencriptación durante el siguiente periodo criptográfico.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación.
ecmId: uint	Número de identificación del ECM asignado por el **Microservidor** para la sincronización de mensajes de datos.
length: uint	Longitud en bytes del parámetro **ecm**; el ecm tiene un único formato de sección.
ecm: byte[]	Mensaje ECM a insertar en el siguiente periodo criptográfico.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta, la sesión utiliza el *modo sincronización* **SyncTs**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** está listo para insertar el siguiente ECM.

Información de la Semántica:

- El **Anfitrión ECI** insertará el ECM en el flujo de transporte dentro de cierto periodo de tiempo tras la recepción del mensaje. En [b-UIT-T J Supl. 7] se proponen valores del intervalo de tiempo. El ECM se repetirá a intervalos razonables (como se define en [ISO/CEI 13818-1-1]. El PID del ECM será un PID libre, generado por el **Anfitrión ECI**.
- El **Anfitrión ECI** puede actualizar cualquier información de la PMT en el flujo que refleje el PID del EMC o, en otro caso, enviará la información del PID del EMC para permitir al **Microcliente** obtener más adelante la información de desencriptación necesaria.
- Cuando se modifica un elemento del contenido y/o se produce algún otro cambio en la encriptación de capa superior, el **Microservidor** podrá enviar dos mensajes ECM sucesivos pero distintos para el siguiente periodo criptográfico. El **Anfitrión ECI** insertará como mínimo el último para el resto del periodo. En el modo desfase temporal/almacenamiento insertará el último ECM durante todo el periodo criptográfico.

9.7.2.5.18 Mensaje reqEncrFileData

H→C reqEncrFileData(ushort **mh**, byte **syncKid[**MaxUuidLen], uint **datalength**, byte **data[])**
C→H resEncrFileData(ushort **mh)**

- Este mensaje permite al **Microservidor** proporcionar al **Anfitrión ECI** un mensaje para reenviar al **Microcliente Objetivo** del **Asa de Medios** para la desencriptación, incluida la información de sincronizaciones relacionada con el KeyID.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de encriptación.
syncKid [MaxUuidLen]: byte	KeyId que se utilizará para encriptar el siguiente "fragmento" del fichero para el que el **Microcliente** necesita los datos asociados para la desencriptación.
datalength: uint	Longitud en bytes de los datos.
data[]: byte	Datos destinados al **Microcliente** con fines de desencriptación. El formato de los datos es opaco si el modo del formato de datos es **OfflineDataMode** y es una caja PSSH para su inclusión en una caja MOOV o MOOF ISOBMFF si el modo del formato de datos es **OfflineIsobmffMode**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta, la sesión está en *modo reencriptación* **OfflineStream** o **OfflineStorage** y en *modo sincronización* **SyncFile**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** está listo para recibir el siguiente mensaje de datos.

Información de la Semántica:

- El **Anfitrión ECI** debe asegurar que a cualquier **Microcliente Objetivo** se proporcionan los datos conforme a los requisitos de sincronización junto con el contenido encriptado.
- El **Anfitrión ECI** creará un fichero ISOBMFF válido incluyendo la caja PSSH o bien garantizará que los datos se transfieren junto con el contenido del fichero al **Microcliente** y se proporcionan al **Microcliente** con arreglo a los requisitos de sincronización de datos.
- El **Anfitrión ECI** almacenará adecuadamente en memora intermedia los datos del mensaje **reqEncrMsgRecv** (como datos asociados al contenido). Los valores de los requisitos del tiempo de **Respuesta** se definen en [b-UIT-T J Supl. 7].
- El **Microservidor** puede producir uno o más mensajes de datos antes del inicio de la **Sesión de reencriptación** cuando se encuentra en modo **OfflineStream**.
- El **Microservidor** producirá al menos un mensaje de datos al final de la sesión de encriptación en modo **OfflineStorage**. Normalmente, el **Microcliente** debe procesar este mensaje de datos antes que cualquier contenido.

9.7.2.5.19 Códigos de error de la API de reencriptación

Cuadro 9.7.2.5.19-1 – Códigos de error de la API de reencriptación

Nombre	Valor	Descripción
ErrEncrAuthInconclusive	1	La autenticación solo se ha procesado parcialmente y no ha sido concluyente, pero no se ha producido ningún error.
ErrEncrAuthFail	-256	No ha sido posible identificar el estado de la autenticación parental del elemento de contenido aunque esta se realizó correctamente.
ErrEncrUserMissing	-257	El **Usuario** no proporciona al **Microservidor** entradas de datos esenciales para realizar o continuar realizando la reencriptación del contenido.
ErrEncrCardMissing	-258	La **Tarjeta inteligente** es necesaria para la reencriptación pero no pudo reconectarse con éxito y participar en la reencriptación del contenido.
EnrEncrServiceMissing	-259	No hay disponible en un tiempo razonable un servicio (externo al **CPE**) de apoyo al **Microservidor** en una sesión de desencriptación.
ErrEncrResourceMissing	-260	No hay disponible un recurso no especificado interno al **CPE** para procesar y reencriptar el contenido.

Cuadro 9.7.2.5.19-1 – Códigos de error de la API de reencriptación

Nombre	Valor	Descripción
ErrEncrMmiMissing	-261	Se requiere el acceso del **Microservidor** al MMI, pero no está disponible.
ErrEncrClientAuthError	-262	El **Microservidor** fracasa en la autenticación del **Microcliente Objetivo**.
ErrEncrIpNone	-263	**El Microservidor** no puede proporcionar una dirección IP para las comunicaciones del **Microcliente.**

9.7.2.6 API de desencriptación de Microcliente

9.7.2.6.1 Generalidades

La API de desencriptación de **Microcliente** permite a este desencriptar contenidos de un **Microservidor**.

La fase de descubrimiento permite a un **Microcliente** publicar los objetivos de desencriptación para los cuales puede ofrecer servicios de desencriptación y suministrar las credenciales que permiten a un **Microservidor** crear como objetivo una conexión autenticada con el mismo.

El **Microcliente** debe soportar modos de desencriptación que abarquen los modos de encriptación ofrecidos por su **Microservidor** complementario. Basado en alguno de los modos habitualmente soportados, el **Microcliente** puede desencriptar servicios: ello se basa en la API de desencriptación común.

Forman parte de esta API mensajes de apoyo adicionales de transferencia de datos para la desencriptación en ambos sentidos entre **Microservidor** y **Microcliente** para los diversos modos.

Los mensajes de la API de desencriptación del **Microcliente** figuran en el Cuadro 9.7.2.6.1-1.

Cuadro 9.7.2.6.1-1 – Mensajes de la API de desencriptación

Mensaje	Tipo	Dir.	Etiqueta	Descripción
setDcrModes	set	C→H	0x0	El **Microcliente** informa al **Anfitrión ECI** de los modos que soporta (modo encriptación, modo formato de datos y modo sincronización).
reqDcrTargets	A	H→C	0x1	El **Anfitrión ECI** solicita al **Microcliente** que proporcione los objetivos de encriptación para los que pueda desencriptar servicios.
reqDcrTargetCred	A	H→C	0x2	El **Anfitrión ECI** solicita al **Cliente ECI** que proporcione los datos de inicialización para una conexión de **Microservidor** normalmente utilizada para la autenticación del objetivo.
reqDcrIpServer	A	C→H	0xA	El **Microcliente** solicita al **Anfitrión ECI** que proporcione la dirección IP del **Microservidor** para comunicaciones adicionales relacionadas con la sesión de **Asa de Medios**.
reqDcrMsgSend	A	C→H	0xB	El **Microcliente** solicita al **Anfitrión ECI** que envíe un mensaje al **Microservidor** de una sesión de **Asa de Medios**.
reqDcrMsgRecv	A	H→C	0xC	El **Anfitrión ECI** proporciona al **Microcliente** un mensaje del **Microservidor** de una sesión de **Asa de Medios**.
reqDcrTsData	A	C→H	0xD	El **Microservidor** proporciona al **Anfitrión ECI** datos que deben enviarse al **Microcliente Objetivo** de un **Asa de Medios** para su desencriptación, incluida la información de sincronización conexa del ECM.
reqEncrFileData	A	C→H	0xF0	El **Microservidor** proporciona al **Anfitrión ECI** un mensaje que debe enviarse al **Microcliente Objetivo** de un **Asa de Medios** para su desencriptación, incluida la información de sincronización conexa del KeyID.

9.7.2.6.2 Mensaje setDcrModes

C→H setDcrModes(EciEncrModes **modes)**

- Este mensaje permite al **Microcliente** informar al **Anfitrión ECI** de los modos que soporta (modo encriptación, modo formato de datos y modo sincronización).

Definición de los parámetros de la Petición:

modes: EciEncrModes	Modos de encriptación que soporta el **Microcliente**. El tipo EciEncrModes se especifica en el Cuadro 9.7.1.5.2-1.

9.7.2.6.3 Mensaje reqDcrTargets

H→C reqDcrTargets() →
C→H resDcrTargets(EncrTarget **target[])**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microcliente** que proporcione los objetivos de encriptación para los que puede realizar la desencriptación.

Definición de los parámetros de la Contestación:

target[]: EncrTarget	Lista de objetivos de encriptación que el **Microservidor** puede autenticar. La definición del tipo de TargetClient (cliente objetivo) se especifica en el Cuadro 9.7.2.5.2-1.

Información de la Semántica:

- El **Anfitrión ECI** puede establecer una correspondencia con potenciales **Microclientes Objetivo** sobre la base del **Objetivo**. La localización de potenciales **Microclientes** candidatos depende de la aplicación y/o del **Anfitrión ECI**.

9.7.2.6.4 Mensaje reqDcrTargetCred

H→C reqDcrTargetsCred(EncrTarget **target) →**
C→H reqDcrTargetsCred(uint **credLen**, byte **cred[])**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microcliente** que proporcione credenciales para la encriptación por parte de un **Microservidor**.

Definición de los parámetros de la Petición:

target: EncrTarget[]	**Objetivo** de encriptación para el que el **Microcliente** tiene que proporcionar las credenciales reales para que un **Microservidor** encripte el contenido.

Definición de los parámetros de la Contestación:

credLen: uint	Longitud en bytes del parámetro **cred**.
cred[]: byte	Credenciales codificadas en un formato específico para el **Microservidor** que encriptará el contenido a desencriptar por el **Microcliente**.

Información de la Semántica:

- Este mensaje permite al **Anfitrión ECI** solicitar al **Microcliente** que proporcione las credenciales correspondientes al parámetro **Objetivo** de forma que un **Microservidor** que reconozca el **Objetivo** pueda encriptar el contenido dirigido al **Microcliente**.

9.7.2.6.5 Mensaje reqDcrIpServer

C→H reqDcrIpServer(ushort **mh) →**
C→H resDcrIpServer(ushort **mh**, Addrinfo **addr)**

- Este mensaje permite al **Microcliente** solicitar al **Anfitrión ECI** que proporcione la dirección IP del **Microservidor** para ulteriores comunicaciones relativas a la sesión de **Asa de Medios**. Los códigos de error conexos se definen en el Cuadro 9.7.2.6.5-1.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de desencriptación para la que se solicita la dirección IP de un **Microservidor** que envía/recibe mensajes.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de desencriptación para la que se proporcionan las direcciones IP de un **Microservidor** que envía/recibe mensajes.
addr: Addrinfo	Protocolo/dirección/puerto IP del **Microservidor** para esta **Asa de Medios**.

Precondiciones a la Petición:

1) La sesión de **Asa de Medios** está abierta en modo **OnlineIpMode**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** conoce que el estado de la encriptación de la sesión es no iniciado.

Información de la Semántica:

- El intercambio IP entre **Microcliente** y **Microservidor** es específico del **Sistema microDRM**. Incluye la elección del protocolo y de cualquier convenio para la terminación de una conexión o el intercambio en una sesión de flujo de contenido.
- Este mensaje puede generarse en una sesión de **Asa de Medios** en la que el proceso de reencriptación aún no haya comenzado.

Cuadro 9.7.2.6.5-1 – Códigos de error de reqDcrIpServer

Nombre	Descripción
ErrDcrIpNone	Véase el Cuadro 9.7.2.6.10-1.

9.7.2.6.6 Mensaje reqDcrMsgSend

C→H reqDcrMsgSend(ushort **mh**, uint **length**, byte **msg[]) →**
C→H resDcrMsgSend(ushort **mh)**

- Este mensaje permite al **Microcliente** solicitar al **Anfitrión ECI** el envío de un mensaje al **Microservidor Objetivo** asociado al **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de desencriptación para la que debe enviarse un mensaje al **Microservidor**.
length: uint	Longitud en bytes del campo **msg**.
msg[]: byte	Mensaje a enviar al **Microservidor**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación.

Precondiciones a la Petición:

1) La sesión de **Asa de Medios** está abierta en modo **OnlineMsgMode**.

Precondiciones a la Contestación:

1) El mensaje se ha enviado al **Microservidor**; el **Anfitrión ECI** está listo para aceptar una nueva **reqDcrMsgSend**.

Información de la Semántica:

- El **Anfitrión ECI** podrá procesar y enviar al menos un mensaje al **Microservidor** en cualquier momento. Los mensajes deben entregarse en orden. El **Anfitrión ECI** no está obligado a proporcionar memoria intermedia específica con capacidad simultánea para más de una petición **reqDcrMsgSend** pendiente. Una implementación segura del **Microcliente** debería utilizar **resDcrMsgSend** como mecanismo de toma de contacto del flujo de control.
- La fiabilidad del servicio de envío de mensajes entre el **Microservidor** y el **Microcliente** es la definida para **reqEncrMsgSend** en la cláusula 9.7.2.5.14.

9.7.2.6.7 Mensaje reqDcrMsgRecv

H→C reqDcrMsgRecv(ushort **mh**, uint **length**, byte **msg[]) →**
C→H resDcrMsgRecv(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** proporcionar al **Microcliente** un mensaje procedente del **Microservidor Objetivo**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de desencriptación para la que el **Microcliente** obtiene un mensaje procedente del **Microservidor**.
length: uint	Longitud en bytes del campo **msg**.
msg[]: byte	Mensaje a recibir del **Microservidor.**

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de desencriptación.

Precondiciones a la Petición:

1) La sesión del **Asa de Medios** está abierta en modo **OnlineMsgMode**.

Precondiciones a la Contestación:

1) El mensaje ha sido procesado por el **Microcliente** y este está listo para aceptar una nueva **reqDcrMsgRecv**.

Información de la Semántica:

- El **Microcliente** procesará como mínimo un mensaje cada vez. El **Microcliente** no está obligado a proporcionar memoria intermedia específica con capacidad simultánea para más de una petición **reqDcrMsgSend** pendiente, aunque debería hacerse cargo y estar listo para procesar un mensaje subsiguiente manteniendo sus demás necesidades en términos de capacidad de respuesta. Una implementación segura del **Anfitrión ECI** debería utilizar **resDcrMsgRecv** como mecanismo de toma de contacto del flujo de control.
- La fiabilidad del servicio de envío de mensajes entre el **Microcliente** y el **Microservidor** es la definida para **reqEncrMsgSend** en la cláusula 9.7.2.5.14.

9.7.2.6.8 Mensaje reqDcrTsData

H→C reqDcrTsData(ushort **mh**, uint **length**, byte **msg[]) →**
C→H resDcrTsData(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** proporcionar al **Microcliente** datos que serán necesarios en un (próximo) futuro para desencriptar el contenido en el **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de desencriptación.
length: uint	Longitud en bytes del mensaje a enviar.
msg[]: byte	Mensaje a enviar al **Microcliente**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de desencriptación.

Precondiciones a la Petición:

1) La sesión de **Asa de Medios** está abierta, la sesión está en *modo reencriptación* **OfflineStream** u **OfflineStorage**, utiliza el *modo formato de datos* **OfflineDataMode** y el *modo sincronización* **SyncTs**.

Precondiciones a la Contestación:

1) El **Anfitrión ECI** está listo para recibir un nuevo mensaje de datos.

Información de la Semántica:

- El **Anfitrión ECI** debería asegurar que se facilitan al **Microcliente** datos acordes con los requisitos de sincronización proporcionados por el **Microservidor** junto con el contenido encriptado a desencriptar.
- El **Microcliente** recibirá como máximo un mensaje de datos al inicio de la sesión de desencriptación en modo **OfflineStorage**. Por lo tanto, se aplica el modo "almacenamiento fuera de línea".

9.7.2.6.9 Mensaje reqDcrFileData

H→C reqDcrFileData(ushort **mh**, uint **datalength**, byte **data[])**
C→H resDcrFileData(ushort **mh)**

- Este mensaje permite que el **Anfitrión ECI** proporcione al **Microcliente** datos del **Microservidor Objetivo** necesarios para desencriptar contenidos del **Asa de Medios**.

Definición de los parámetros de la Petición:

mh: ushort	**Asa de Medios** para la sesión de desencriptación.
datalength: uint	Longitud en bytes de los datos.
data[]: byte	Datos dirigidos al **Microcliente** para su desencriptación. El formato de los datos es opaco si el modo formato de datos es **OfflineDataMode** y es una Caja PSSH para su inclusión en una caja MOOV o MOOF ISOBMFF si el modo formato de datos es **OfflineIsobmffMode**.

Definición de los parámetros de la Contestación:

mh: ushort	**Asa de Medios** para la sesión de encriptación.

Precondiciones a la Petición:

1) La sesión de **Asa de Medios** está abierta, la sesión está en modo reencriptación **OfflineStream** o **OfflineStorage** y en modo sincronización **SyncFile**.

Precondiciones a la Contestación:

1) El **Microcliente** está listo para recibir un nuevo mensaje de datos.

Información de la Semántica:

- El **Anfitrión ECI** debe asegurar que se facilitan al **Microcliente** datos coherentes con los requisitos de sincronización junto con el contenido encriptado.

- El **Anfitrión ECI** puede extraer una caja PSSH de un fichero ISOMFF válido proporcionado al **Microcliente** alineado con los requisitos de sincronización de datos para la decodificación de ficheros ISOBMFF.
- El **Anfitrión ECI** proporcionará como máximo un mensaje de datos al final de la sesión de encriptación en modo **OfflineStorage**. Normalmente este mensaje de datos tiene que ser procesado por el **Microcliente** antes que cualquier contenido.

9.7.2.6.10 Códigos de error de la API de desencriptación del Microcliente

Los códigos de error de la API de desencriptación del Microcliente figuran en el Cuadro 9.7.2.6.10-1.

Cuadro 9.7.2.6.10-1 Códigos de error la API de desencriptación del Microcliente

Nombre	Valor	Descripción
ErrDcrIpNone	-256	El **Anfitrión ECI** no dispone de dirección/puerto IP para la comunicación con el **Microservidor**.

9.8 Conjunto de las API de recursos asociados a las propiedades del contenido

9.8.1 Lista de las API definidas en la cláusula 9.8

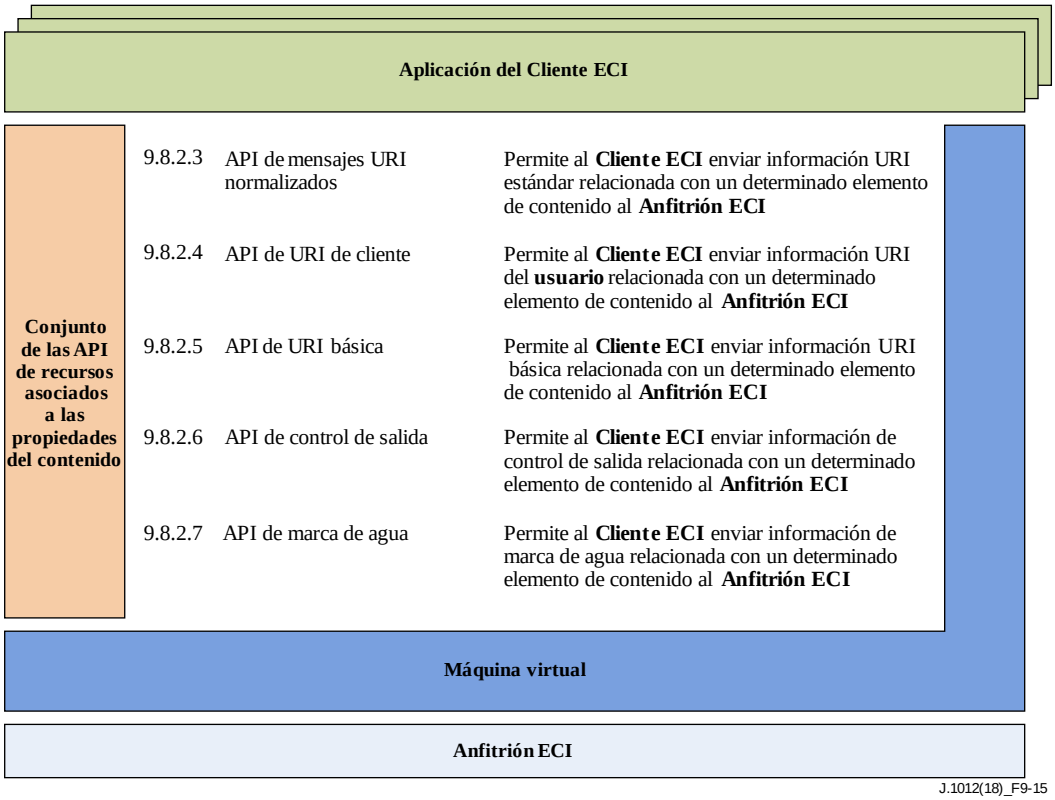


Figura 9.8.1-1 – Representación esquemática de las API definidas en la cláusula 9.8

En el Cuadro 9.8.1-1 figuran las API incluidas en la cláusula 9.8 y la Figura 9.8.1-1 ilustra la ubicación de las API definidas en la cláusula 9.8 con la **arquitectura ECI**.

Cuadro 9.8.1-1 – Conjunto de las API de recursos relativos a la protección de contenido

Cláusula	Nombre de la API	Descripción
9.8.2.3	API de mensaje URI normalizado	Permite al **Cliente ECI** enviar información URI estándar relacionada con un determinado elemento de contenido al **Anfitrión ECI** y viceversa.
9.8.2.4	API de URI de cliente	Permite al **Cliente ECI** enviar información URI de **Usuario** relacionada con un determinado elemento de contenido al **Anfitrión ECI** y viceversa.
9.8.2.5	API de URI básico	Permite al **Cliente ECI** enviar información URI básica relacionada con un determinado elemento de contenido al **Anfitrión ECI** y viceversa.
9.8.2.6	API de control de salida	Permite a un **Cliente ECI** enviar información de control de salida relacionada con un determinado elemento de contenido al **Anfitrión ECI** y viceversa.
9.8.2.7	API de filigrana	Permite al **Cliente ECI** enviar información de filigrana relacionada con un determinado elemento de contenido al **Anfitrión ECI** y viceversa.
9.8.2.8	API de control parental	Permite a un **Cliente ECI** enviar información sobre obligaciones del control parental asociada a un determinado elemento de contenido al **Anfitrión ECI**.
9.8.2.9	API de sincronización de propiedades del contenido	Permita la sincronización de varios cambios en las propiedades del contenido.
9.8.2.10	API de autenticación parental	Permite a un **Cliente ECI** delegar la autenticación parental a una función de autenticación parental estándar en el **Anfitrión ECI**.
9.8.2.11	API de delegación de la autenticación parental	Permite a un **Cliente ECI** cancelar una petición de autenticación parental delegada.
9.8.2.12	API de control de protección	Permite al **Cliente ECI** proporcionar control específico del **Operador de Plataforma** sobre los Sistema de protección de salida.

9.8.2 Conjunto de las API de acceso a derechos de uso y recursos del control parental

9.8.2.1 Introducción

Esta cláusula relativa a las API de **Cliente/Anfitrión ECI** permite al **Cliente ECI** establecer los derechos y condiciones aplicables al contenido desencriptado de una forma segura.

La API de derechos y condiciones especifica los aspectos siguientes:

- URI (Información de derechos de uso) estándar: generada por el **Cliente ECI** y utilizada por el **Anfitrión ECI** para controlar las aplicaciones del contenido a productos y aplicaciones normalizados de la industria.
- URI básica: generada por el **Cliente ECI** y utilizada por la Seguridad avanzada y el subsistema hardware del **Anfitrión ECI** para establecer los derechos de uso básicos relativos al contenido. Permite al **Cliente ECI** utilizar una protección hardware robusta para las propiedades de derechos básicos que son necesarias con relación al contenido.
- Control de salida: permite al **Cliente ECI** bloquear selectivamente salidas que podrían estar activas en las condiciones de la URI pero que, no obstante, su uso se considera inadecuado desde una perspectiva de derechos.
- Control de la filigrana impulsada por el **Anfitrión ECI**: permite al **Cliente ECI** marcar contenidos salientes con marcas especificadas por el **Cliente ECI** mediante un sistema de marcaje de agua residente en el **CPE**.
- Condiciones de control parental: permiten al **Cliente ECI** enviar los requisitos para la autenticación de un Padre a fin de dar acceso al contenido al sistema de protección al cual se exporta el contenido.
- Sincronización de las propiedades del contenido: permite que se produzcan varios cambios simultáneos en las propiedades del contenido de forma que se identifiquen como tales.

- Función de autenticación parental: puede realizarla un **Cliente ECI** o ser delegada a una función estándar de la industria en el **Anfitrión ECI**. A su vez, el **Anfitrión ECI** puede seleccionar un **Cliente ECI** específico para realizar la autenticación parental en su nombre. Las opciones de delegación permiten una única autenticación parental en varios **Clientes ECI** y el **Anfitrión ECI**.

La aplicación de nuevas propiedades de derechos está vinculada de forma segura a la aplicación de una nueva palabra de control para desaleatorizar el contenido. Ello garantiza que los derechos se apliquen al contenido al que están asociados.

Las API de propiedades del contenido tiene mensajes *set* (fijar) y mensajes *get* (obtener). Los **Clientes ECI** que desencriptan el contenido utilizan el mensaje *set* para señalizar las propiedades del contenido asociadas con la siguiente palabra de control calculada. Los **Microservidores** que reencriptan el contenido utilizan la función *get* para adquirir las propiedades del contenido entrante al objeto de construir la autenticación adecuada y los datos de señalización para identificar las propiedades del contenido reencriptado.

La versión de la API identificada como parte de la API de descubrimiento armoniza de manera efectiva la versión de las propiedades del contenido utilizadas.

El contexto del **Asa de Medios** del **Anfitrión ECI** mantendrá como mínimo dos valores para distintas secciones de contenido de cada propiedad del contenido. Por lo tanto, para la desencriptación basada en ficheros mantendrá al menos dos secciones de contenidos, cada una con un KeyID diferente para cada propiedad de contenido. En el Cuadro 9.8.2.1-1 figuran las funciones de la API. Las funciones de la API relativas a los derechos se agrupan en diversas API a fin de permitir una gestión independiente de las versiones.

Cuadro 9.8.2.1-1 – Lista de mensajes de la API de derechos de uso y control parental

API	Mensaje	Tipo	Dir.	Etiqueta	Descripción
ApiStdUri	setDcrStdUri	set	C→H	0x0	Fija la URI estándar para el contenido a desaleatorizar.
ApiStdUri	getEncrStdUri	get	C→H	0x1	Obtiene la URI estándar para el contenido a reencriptar.
ApiCustUri	setDcrCustUri	set	C→H	0x0	Fija la URI a medida para el contenido a desaleatorizar.
ApiCustUri	getEncrCustUri	get	C→H	0x1	Obtiene la URI a medida para el contenido a reencriptar.
ApiBasicUri	setDcrBasicUri	set	C→H	0x0	Fija la URI básica para el contenido a desaleatorizar.
ApiBasicUri	getEncrBasicUri	get	C→H	0x1	Obtiene la URI básica para el contenido a reencriptar.
ApiOC	setDcrOutputCtl	set	C→H	0x0	Fija restricciones de control de salida para el contenido a desaleatorizar.
ApOC	getEncrOutputCtrl	get	C→H	0x1	Obtiene restricciones de control de salida para el contenido a reencriptar.
ApiDcrMark	getDcrMarkSyst	get	H→C	0x0	Obtiene los sistemas de marcaje permitidos.
ApiDcrMark	setDcrMarkMeta	set	C→H	0x1	Fija un valor del control del sistema de marcaje.
ApiDcrMark	getDcrMarkMeta	get	H→C	0x2	Lee una propiedad del sistema de marcaje.
ApiDcrMark	setDcrMarkBasic	set	C→H	0x3	Fija la carga útil del marcaje básico para el contenido a desaleatorizar.
ApiDcrMark	setDcrMarkExt	set	C→H	0x4	Fija la carga útil del marcaje básico para el contenido a desaleatorizar.
ApiPar	setDcrParCtl	set	C→H	0x0	Fija las condiciones del control parental para el contenido a desaleatorizar.
ApiPar	getEncrParCtrl	get	C→H	0x1	Obtiene las condiciones del control parental para el contenido a desaleatorizar.

Cuadro 9.8.2.1-1 – Lista de mensajes de la API de derechos de uso y control parental

API	Mensaje	Tipo	Dir.	Etiqueta	Descripción
ApiCpSync	setCpSync	set	C→H	0x0	El **Cliente ECI** señaliza que el conjunto actual de propiedades del contenido es consistente y puede aplicarse al contenido a desaleatorizar utilizando la palabra de control siguiente.
ApiCpSync	reqCpChange	req	H→C	0x1	El **Anfitrión ECI** señaliza que se va a producir un cambio en las propiedades del contenido a reencriptar.
ApiParAuth	reqParAuthChk	req	C→H	0x0	Solicita al **Anfitrión ECI** que realice una autenticación parental en nombre del **Cliente ECI**.
ApiParAuth	reqParAuthChkCan	req	C→H	0x1	Cancela una petición previa al Anfitrión de autenticación parental.
ApiParAuth	reqParAuthCid	req	H→C	0x2	Solicita autorización parental mediante código pin para un (futuro) elemento de contenido a decodificar. Puede activar un diálogo de autenticación parental.
ApiParAuthDel	reqParAuthDel	req	H→C	0x0	El **Anfitrión ECI** delega una autenticación parental a un **Cliente ECI**.
ApiParAuthDel	reqParAuthDelCan	req	H→C	0x1	El **Anfitrión ECI** cancela una petición de autenticación parental previa al **Cliente ECI**.
ApiProtCtrl	getProtSystCtrl	get	C→H	0x0	El **Cliente ECI** obtiene del **Anfitrión ECI** la lista de sistemas de protección de salida y su soporte para SRM (mensajes de capacidad de renovación del sistema) y servicios de bloqueo de ID de dispositivos.
ApiProtCtrl	reqSrmMsg	req	C→H	0x1	El **Cliente ECI** proporciona un SRM al sistema de protección de salida.
ApProtCtrl	reqInfoDevId	req	H→C	0x2	El **Anfitrión ECI** proporciona el ID del dispositivo al que el sistema de protección de salida proporciona contenido protegido en una sesión de desencriptación.
ApiProtCtrl	reqBlockDevId	req	C→H	0x3	El **Cliente ECI** proporciona el ID del dispositivo al que no se le proporcionará ningún contenido mediante un sistema de protección de salida en una sesión de desencriptación.
ApProtCtrl	setBlockProtSyst	set	C→H	0x4	El **cliente de ECI** indica que el sistema de protección se considera inadecuado para proteger el contenido de la sesión de desencriptación.

9.8.2.2 Aspectos de seguridad y sincronización

La especificación **ECI** permite que la información sobre las propiedades del contenido arriba indicada sea autenticada por el **Anfitrión ECI** a fin de impedir la manipulación no autorizada de esa información. Este mecanismo también garantiza que se aplican los valores correctos de los parámetros al contenido al que están asociados. Esto se define en [UIT-T J.1014].

En relación con la información sobre las propiedades del contenido, el **Anfitrión ECI** puede facilitar la autenticación de la información sobre los derechos en nombre del **Cliente ECI** utilizando claves en el Bloque de Seguridad avanzada, garantizando el máximo nivel de integridad posible en la autenticación. Es decisión de los **Clientes ECI** utilizar los servicios AS del **Anfitrión ECI** para este fin. Esto también se define [UIT-T J.1014].

En el caso de que las propiedades del contenido requieran la aplicación de propiedades específicas de protección de salida en una salida, pero dichas propiedades de protección de salida (o versiones más seguras o limitadas de las mismas) no pueda proporcionarlas el **Anfitrión ECI**, éste no dará salida al contenido y suministrará un mensaje adecuado al **usuario**. En el régimen de cumplimiento del **Ecosistema ECI** se proporcionará información más detallada al respecto.

9.8.2.3 API de mensajes de URI estándar

9.8.2.3.1 Mensaje setDcrStdUri

C→H setDcrStdUri(ushort **mh**, byte **keyId**[MaxUuidLen], StdUri **stdUri)**

- Este mensaje establece que el valor de la URI estándar asociada a **keyId** sea **uri**.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID a como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS al objeto de indicar su aplicabilidad a la CW siguiente.
stdUri: StdUri	La URI estándar para contenidos se define en el Cuadro 9.8.2.3.1-1. La semántica de los campos corresponde a las definidas en [ETSI TS 103 205] y [b-CI Plus].

Cuadro 9.8.2.3.1-1 – Especificación del tipo de URI estándar

```
typedef struct StdUri {

    uint MajorVersion: 4;
    uint tmc: 1;   /* trick_mode_control_info en [CI+ v1.4] */
    unit reserved1: 3;
    uint aps: 2; /* aps_copy_control_info en [CI+ v1.4] */
    uint emi: 2; /* emi_copy_control_info en [CI+ v1.4] */
    uint ict: 1; /* ict_copy_control_info en [CI+ v1.4] */
    uint rct: 1; /* rct_copy_control_info en [CI+ v1.4] */
    uint reserved2: 1; /* reserved bit */
    uint dot: 1; /* dot_copy_control_info en [CI+ v1.4] */
    uint rl: 8;     /* rl_copy_control_info en [CI+ v1.4] */

} StdUri;

Serán de aplicación las normas siguientes (las expresiones sobre el campo se
evaluarán para determinar si son Verdadero) conforme a [CI+v1.4]
emi == 0b00 || rct == 0b0
emi == 0b11 || (dot == 0b0 && rl == 0x00)
emi == 0b01 || tmc == 0b0

El valor 0x03 del campo protocol_version se define para el caso anterior; otros
valores quedan reservados para uso futuro.
```

Semántica de los campos de StdUri:

MajorVersion: uint: 4	Versión principal de esta URI estándar. Los **Clientes ECI** fijarán el valor de MajorVersion a 0b0000. Los **Anfitriones ECI** implementarán todas las versiones hasta su nivel de cumplimiento para este campo e interpretarán cualquier valor elevado como una URI no implementada y por lo tanto sin la aplicación de ningún derecho de uso.
reserved1: unit: 3	Bits reservados. El **Cliente ECI** los fijará a 0b000 y serán ignorados por los **Anfitriones ECI** que sean conformes con esta versión de stdUri.
reserved2: unit: 3	Bit reservado. El **Cliente ECI** lo fijará a 0b0 y será ignorado por los **Anfitriones ECI** que sean conformes con esta versión de stdUri.
Otros campos	En la anterior definición de estructura, la semántica es la definida para los campos indicados de la URI CI Plus v1.4 [ETSI TS 103 205].

Información de la Semántica:

- Para el modo desaleatorización del flujo de transporte, se aplicará la URI al contenido a decodificar, aplicando las claves a la siguiente clave de desencriptación. En la cláusula 8.2.4.7 de [UIT-T J.1014] se definen los detalles para el cálculo de la clave de desencriptación.
- El **Cliente ECI** estará en modo desencriptación.

9.8.2.3.2 Mensaje getEncrStdUri

C→H StdUri getEncrStdUri(ushort **mh**, byte **keyId**[MaxUuidLen]**)**

- Este mensaje establece la URI estándar para un contenido posterior.

Definición de propiedades:

- La URI estándar se define en el Cuadro 9.8.2.3.1-1.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId: byte[MaxUuidLen]	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.

9.8.2.4 API de URI de cliente

9.8.2.4.1 Mensaje setDcrCustUri

C→H setDcrCustUri(ushort **mh**, byte **keyId**[MaxUuidLen], unit **custUriLen**, byte ***custUri)**

- Este mensaje establece que el valor de la URI a medida asociada a **keyId** sea **uri**.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId: byte[MaxUuidLen]t	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
custUriLen: unit	Longitud en bytes del campo URI a medida.
custUri: byte *	En el Cuadro 9.8.2.4.1-1 se define la URI a medida para el contenido. Los bytes 0 y 1 actuarán como msB, y lsB del formato de la URI a medida. Todos los valores de los bytes 0 y 1 están reservados excepto 0x80 y 0x00 que tendrán un significado específico de la aplicación para los bytes que siguen.

Cuadro 9.8.2.4.1-1 – Especificación de tipo de URI a medida

Nombre	Valor de bytes 0, 1	Descripción
CustUriPrivate	0x80, 0x00	El significado de los bytes que siguen al byte 1 es privado. La interpretación adecuada del resto del campo se define mediante otra comunicación entre el **Cliente ECI** y el **Microservidor** o el sistema de protección.
RFU	Otros	Reservado para uso futuro.

Información de la Semántica:

- Para el modo desaleatorización del flujo de transporte, la URI se aplicará al contenido a decodificar, aplicando las claves a la siguiente **reqDcrTsDescrKey**.
- Se permite fijar un máximo de 4 URI diferentes a medida para una palabra de control.
- El **Cliente ECI** estará en modo desencriptación.

9.8.2.4.2 Mensaje getEncrCustUri

C→H custUri getEncrCustUri(ushort **mh**, byte **keyId**[MaxUuidLen], unit **custUriMaxLen)**

- Este mensaje obtiene la URI a medida para un próximo contenido.

Definición de propiedades:

- La URI a medida se define en el Cuadro 9.9.1-1.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId: byte[MaxUuidLen]	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 transporta 0x00 (par) o 0x01 (impar) en flujos de formato TS para indicar su aplicabilidad a la CW siguiente.
custUriMaxLen: uint	Longitud máxima (en bytes) del resultado de la URI a medida: cualquier contenido adicional será truncado.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.

9.8.2.5 API de URI básica

9.8.2.5.1 Mensaje setDcrBasicUri

C→H setDcrBasicUri(ushort **mh**, byte **keyId**[MaxUuidLen], BasicUri **basicUri)**

- Este mensaje establece que el valor de la URI básica asociada con **keyId** sea **basicUri**. La URI básica proporciona una gestión de derechos simplificada pero muy robusta para contenido a desencriptar.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
basicUri: BasicUri	La URI básica para el contenido se define en el Cuadro 9.8.2.5.1-1. La semántica de los campos es la definida en [ETSI TS 103 205].

Cuadro 9.8.2.5.1-1 – Especificación del tipo de URI básica

```
typdef byte  BasicUri;
```

Nombre	Bits	Descripción
BasicUriVersion	7	Versión principal de URI básica. Si el **Anfitrión ECI** no ha implementado la versión, el **Cliente ECI**, no permitirá la desencriptación y uso del contenido. El valor 0b0 define la versión 0. Los demás valores quedan reservados y no están permitidos.
BasicUriV0_0Ext	2..6	Reservado para uso futuro, no se utiliza en v0.0. El único valor definido para este campo es 0b00000. No se permiten otros valores. Los **Anfitriones ECI** que solo implementen la URI básica v0.0 ignorarán los valores de este campo: es decir, puede utilizarse en futuras ampliaciones retrocompatibles de v0.0; por ejemplo, con una mayor flexibilidad en el control de derechos v0.0.
BasicUriV0_0	0,1	Versión 0.0 de la URI básica. Los valores y significados de este campo se definen en el Cuadro 9.8.2.5.1-2.

Cuadro 9.8.2.5.1-2 – Definición de la v0.0 de la URI básica

Nombre	Valor	Descripción
NoBasicProtection	0b00	La URI básica no permite el control de derechos.
RedistributionProtected	0b01	La encriptación estará activada, la prevención de reproducción desactivada.
ViewOnly	0b10	La encriptación estará activada, la prevención de reproducción activada.
ViewOnlyStrict	0b11	La encriptación estará activada, la prevención de reproducción activada, la salida restringida a salidas específicamente cualificadas (seguras).

Información de la Semántica:

- Para el modo desaleatorización del flujo de transporte, la URI aplicará al contenido a decodificar con las claves aplicadas a la siguiente **reqDcrTsDescrKey**.
- La URI básica permite que el **Cliente ECI** controle la implementación de derechos con el máximo nivel de robustez que permite el **Anfitrión ECI**. Ofrece el control de dos mecanismos de protección: encriptación, que garantiza que el contenido siempre se aleatoriza, cualquiera que sea la salida o el medio de almacenamiento, y prevención de reproducción, que garantiza que el contenido encriptado sólo puede ser desaleatorizado en una conexión activa (es decir, no puede ser almacenado). Para más información véase [UIT-T J.1015].
- El **Cliente ECI** estará en modo desencriptación.

9.8.2.5.2 Mensaje getEncrBasicUri

C→H BasicUri getEncrBasicUri (ushort **mh**, byte **keyId** [MaxUuidLen]**)**

- Este mensaje obtiene la URI básica para contenido que sigue a continuación.

Definición de propiedades:

- La URI básica se define en el Cuadro 9.8.2.5.1-1.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.

9.8.2.6 API de control de salida

9.8.2.6.1 Mensaje setDcrOutputCtl

C→H setDcrOutputCtl (ushort **mh**, byte **keyId** [MaxUuidLen], uint **ocVector)**

- Establece que los valores del Control de salida asociados a **keyId** sean los correspondientes a **ocVector**.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
convector: unit	Vector de Control de salida para salidas estándar tal como se define en el Cuadro 9.8.2.6.1-1.

Cuadro 9.8.2.6.1-1 – Especificación del vector de control de salida

Nombre	Bits	Descripción
MajorVersion	7	Versión del parámetro ocVector. El valor 0b0 se define para la versión 1. Cualquier otro valor está reservado y su uso no está permitido. Si un **Anfitrión ECI** que implementa la **versión principa**l 1 recibe un valor distinto de 0xb0 no se permiten salidas.
OcAnyOther	6	Cualquier otra salida del **Anfitrión ECI** no incluida en ningún otro criterio de cualificación de salida enumerado más abajo. Si el valor es 0b0, en esas salidas podrá haber algún dato de salida, si el valor es 0b1 no se permitirá dato de salida alguno. ***El valor de este bit cambia la codificación de los campos indicados más abajo***. Si el valor es 0b0, las restricciones de salida serán las enumeradas más abajo. Si el valor es 0b1, la codificación se invertirá a nivel de bit. Es decir, si OcAnyOther==0b1 y OcIP==0b1 se permiten salidas en la conexión IP. Véase la Nota 2.
OcIP	0	Si el valor es 0b0 se permiten salidas en cualquier conexión IP, pero no cuando el valor es 0b1.
OcUSB	1	Si el valor es 0b0 se permiten salidas en cualquier conexión USB, pero no cuando el valor es 0b1. La Precondición para ello es que el contenido desencriptado no esté protegido por ningún sistema de protección de salida reconocido **ECI** y/o **Sistema microDRM ECI** bajo el control del **Cliente ECI** de desencriptación.
OcDtcpIp	2,3	Si el valor es 0b0 se permiten salidas en una conexión protegida DTCP-IP, no así cuando el valor sea 0b1.
OcHdcp	3,4	Cualquier salida protegida HDCP. Si OcAnyOther es 0b0: • valor 0b00: se permiten salidas con protección HDCP; • valor 0b01: si la versión de HDCP es inferior a 2.2 no se permiten salidas; si la versión de HDCP 2.2 o superior se permiten salidas; • valor 0b10: reservado; este valor no está permitido. Los **Anfitriones ECI** interpretarán que el valor es 0b11; • valor 0b11: no se permite ningún dato de salida con protección HDCP. Si OcAnyOther es 0b1: • valor 0b00: no se permitirán salidas HDCP; • valor 0b01: reservado, los **Anfitriones ECI** interpretarán que el valor es 0b00; • valor 0b10: si la versión de HDCP es 2.2 o superior se permitirán salidas; si la versión de las salidas HDCP es inferior a 2.2 no se permite salida alguna; • valor 0b11: se permite ningún dato de salida con protección HDCP.
OcWm	5	Si el valor de este bit es 0b1, sólo se permiten salidas del elemento de contenido decodificado con una filigrana insertada por el **CPE** en el elemento de contenido relacionado. Véase la Nota 3.

Cuadro 9.8.2.6.1-1 – Especificación del vector de control de salida

Nombre	Bits	Descripción
NOTA 1 – Los campos **dot** e **ict** de la URI estándar proporcionan de manera efectiva el control de salida analógica. NOTA 2 – OcAnyOther transfiere efectivamente el campo control de salida desde una lista negra de salidas (cuando el valor es 0b0) a una lista blanca de salidas (cuando el valor es 0b1). Si el campo de salida es 0b1 significa que efectivamente se encuentra "en la lista". NOTA 3 – Los sistemas de filigrana adecuados para esta aplicación pueden estar sujetos a aprobación.		

Si se aplican varios campos ocVector a una salida (por ejemplo, una salida IP protegida por DTCP-IP) prevalecer la condición más restrictiva.

Información de la Semántica:

- El **Cliente ECI** estará en modo desencriptación.

9.8.2.6.2 Mensaje getEncrOutputCtrl

C→H uint getEncrOutputCtrl(ushort **mh**, byte **keyId**[MaxUuidLen]**)**

- Este mensaje obtiene el control de salida para contenidos adicionales.

Definición de propiedades:

- El control de salida se define en el Cuadro 9.8.2.6.1-1.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.

9.8.2.7 API de filigrana

9.8.2.7.1 Generalidades

La API de marcaje permite a los **Clientes ECI** descubrir sistemas de marca (filigrana) integrados disponibles a través del **Anfitrión ECI**, y participar en un diálogo de control de "configuración" con esos sistemas. Los sistemas de marcaje pueden dialogar con un número limitado de **Clientes ECI** y sólo pueden marcar simultáneamente un determinado número de sesiones de **Asa de Medios**.

Los sistemas de marcaje pueden interaccionar con **Clientes ECI** autorizados. Esa autorización puede establecerse, entre otras formas, mediante los mensajes setMarkMeta y getMarkMeta con un diálogo de autorización definido por el sistema de marcaje.

Los **Clientes ECI** pueden reservar el acceso a un sistema de marcaje mediante un diálogo de participación de resultados satisfactorios. El **Cliente ECI** (identificado por su id de Cliente ECI) seguirá utilizando el sistema de marcaje hasta que este sea eliminado del **CPE** o hasta que se desentienda del mismo.

9.8.2.7.2 Mensaje getDcrMarkSyst

C→H MarkSystDescr **getDcrMarkSyst()**

- Este mensaje permite al **Cliente ECI** leer los descriptores para los sistemas de marcaje disponibles.

Definición de propiedades:

El tipo de resultado MarkSystDescr será conforme con la definición del Cuadro 9.8.2.7.2-1.

Cuadro 9.8.2.7.2-1 – Definición del tipo MarkSystDescr

```
#define MaxMarkSystDescr 16;

typedef ushort MarkId;  /* ID de marcaje ECI asignado a un sistema de marcaje
*/
// markId values: 0x8xxx se utilizan para sistemas de marcaje patentados.
//                0x0000 indica que no hay sistema de marcaje
//                Los restantes valores están reservados por ECI,
//                la asignación de nuevos ID y su publicación
//                se define en otro documento

typedef struct MarkSystDescrElem {
   MarkID markId;        /* ID del sistema de marcaje */
   uchar  nrClients;     /* número de clientes que aún se soportan */
   uchar markSystFlags  /* campo definido más abajo */
} MarkSystDescr[MaxMarkSystDescr];
// Los sistemas de marcaje disponibles se incluirán como los primeros
// elementos de MarkSystDescr. Los demás elementos usarán markId==0x0000.


// markSystFlags:
//   bit 0 señaliza si es necesaria autorización (0b1)o no(0b0)
//   bit 1 señaliza si se soportan flujos aleatorizados (0b1)o no(0b0)
//   bit 2 señaliza si se soportan varios flujos simultáneos(0b1)o no(0b0)
//   los restantes bits están reservados y serán ignorados por los clientes
//   que sean conformes con la presente Recomendación
```

9.8.2.7.3 Mensaje setDcrMarkMeta

C→H setDcrMarkMeta(MarkID **markId**, uchar **index,** byte **data[32])**

- Este mensaje permite al **Anfitrión ECI** fijar los (meta) datos de control para un sistema de marcaje.

Definición de los parámetros:

markId: MarkID	ID del sistema de marcaje para el que se definen las propiedades.
index: uchar	Subpropiedad a fijar para sistemas de marca.
data[32]: byte	Valor a aplicar a la subpropiedad indicada en el índice.

9.8.2.7.4 Mensaje getDcrMarkMeta

C→H byte[32] getDcrMarkMeta(MarkID **markId**, uchar **index)**

- Este mensaje permite al **Cliente ECI** obtener (meta) datos de control para un sistema de marcaje.

Definición de las propiedades:

- Metadatos para el índice (**index)** de la subpropiedad cuyo ID de marca es **markID**.

Definición de los parámetros:

markId: MarkID	ID del sistema de marcaje para el que debe leerse la definición de la propiedad: el tipo de resultado MarkSystDescr será conforme con la definición del Cuadro 9.8.2.7.4-1.

index: uchar	Subpropiedad del sistema de marcaje a leer.

9.8.2.7.5 Mensaje SetDcrMarkBasic

C→H setDcrMarkBasic(ushort **mH**, byte **keyId**[MaxUuidLen], MarkID, byte **data**[16]**)**

- Este mensaje permite al **Cliente ECI** fijar la utilización de un máximo 128 bits de datos para marcar el contenido a desaleatorizar con la clave designada.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
markId: MarkID	ID del sistema de marcaje.
data[16]: byte	Valor de 128 bits.

9.8.2.7.6 Mensaje SetDcrMarkExt

C→H setDcrMarkExt(ushort **mH**, byte **keyId** [MaxUuidLen], ushort **markId**, uint **dataLen**, byte **data**[]**)**

- Este mensaje permite al **Cliente ECI** fijar una carga útil ampliada para el sistema de marcaje a fin de marcar el contenido a desaleatorizar con la clave designada.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId: byte[MaxUuidLen]	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
markId: ushort	ID del sistema de marcaje a utilizar para marcar el contenido.
datalen: uint	Longitud del campo de datos.
Data[]: byte	Datos de la carga útil para el sistema de marcaje.

9.8.2.8 API de control parental

9.8.2.8.1 Mensaje setDcrParCtl

C→H setDcrParCtl(ushort **mH**, byte **keyId**[MaxUuidLen], ParCond **pC)**

- Este mensaje permite al **Cliente ECI** fijar las condiciones de calificación parental (**pC**) del contenido del mH a desaleatorizar con la clave designada.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la condición de control parental **pC** en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.
pC: ParCond	Condiciones de control parental a aplicar al contenido. En relación con la definición de ParCond véase el Cuadro 9.8.2.8.1-1.

Cuadro 9.8.2.8.1-1 – Especificación del tipo de condición parental

```
typdef struct ParCond {
   byte basicCondition; /* véase el Cuadro 9.8.2.8.1-2 */
   byte extendedQualifier[16];
} ParCond;
```

Cuadro 9.8.2.8.1-2 – Definición de la Condición Parental básica

Nombre	Bits	Descripción
AuthRequired	7	El valor 0b1 significa que es necesaria la autenticación parental antes de presentar contenido. El valor 0b0 significa que la autenticación parental puede ser necesaria en función del extendedQualifier.
ToggleBit	6	Este bit aparece en el flujo para indicar la necesidad de una nueva autenticación parental cuando cambia su valor.
Reserved	4,5	Se fijará en 0b00.
QualifierFormat	0..3	Indica el formato del campo extendedQualifier. El valor 0x0 indica "sin valor", en cuyo caso el campo extendedQualifier tomará el valor todos cero. El valor 0x1 indica que el campo ExtendedQualifier contiene un descriptor de calificación parental DVB definido en [ETSI EN 300 468]. El valor de los demás bytes será cero. La autenticación parental será necesaria incluso si AuthRequired==0b0 cuando la calificación requerida para el país en cuestión exceda el límite fijado por el padre (tal como define la semántica del descriptor de calificación parental DVB). Los valores 0x2..0xF quedan reservados para uso futuro.

Información de la Semántica:

- La **ECI** permite que las condiciones de autenticación de la clasificación parental se transfieran como una obligación junto con el contenido a un sistema de protección del contenido desaleatorizado.
- El **Cliente ECI** estará en modo desencriptación.

9.8.2.8.2 Mensaje getEncrParCtrl

C→H ParCond getEncrParCtrl(ushort **mh**, byte **keyId**[MaxUuidLen]**)**

- Este mensaje permite al **Cliente ECI** obtener la condición del control parental para un contenido adicional.

Definición de propiedades:

- La URI de control parental se define en el Cuadro 9.8.2.8.1-2.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.

9.8.2.9 API de sincronización de las propiedades del control

9.8.2.9.1 Mensaje setCpSync

C→H setCpSync(ushort **mH**, byte **keyId**[MaxUuidLen]**)**

- Este mensaje señaliza al **Anfitrión ECI** que para la siguiente sección de contenido identificada mediante keyId se fijarán las propiedades del contenido mediante las API de URI estándar, URI a medida, URI básica, de control de salida, de filigrana y de control parental.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a decodificar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la condición de control parental **pC** en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El mensaje desencadena que el **Anfitrión ECI** se prepare de forma adecuada para los próximos cambios en las propiedades del contenido. Ello incluirá el envío de un mensaje reqCpChange a cualquier **Microservidor** que tenga una **conexión de importación/exportación** con esa sesión del **Asa de Medios**.
- El **Cliente ECI** estará en modo desencriptación.

9.8.2.9.2 Mensaje reqCpChange

H→C reqCpChange(ushort **mh**, byte **keyId**[MaxUuidLen]**)**

- El mensaje desencadena que al **Microservidor** prepare un cambio en las propiedades del contenido basado en los valores futuros más inmediatos de las propiedades del contenido desencriptado que es reencriptado por el **Microservidor**.

Definición de propiedades:

- La URI de control parental se define en el Cuadro 9.8.2.8.1-2.

Definición de los parámetros:

mH: ushort	**Asa de Medios** del contenido a encriptar.
keyId[MaxUuidLen]: byte	KeyID como UUID con el orden de bytes de la red a la que se aplica la URI en caso de decodificación del formato de fichero, donde el byte 0 es 0x00 (par) o 0x01 (impar) en flujos de formato TS a fin de indicar su aplicabilidad a la CW siguiente.

Información de la Semántica:

- El **Cliente ECI** estará en modo encriptación.
- El **Cliente ECI** obtendrá las propiedades del contenido para un contenido adicional relacionado con el keyId en el flujo desencriptado y preparará una nueva configuración de encriptación para el nuevo contenido (que puede requerir de una nueva CW).

9.8.2.10 API de autenticación parental

9.8.2.10.1 Generalidades

La autenticación para aprobación parental puede ser realizada directamente por un **Cliente ECI** mediante una sesión MMI. Alternativamente, un **Cliente ECI** puede solicitar al **Anfitrión ECI** que realice (o que haya realizado) la autenticación parental, a fin de armonizar la gestión del código pin y mejorar la experiencia de la interfaz de **Usuario** al integrar de forma natural las peticiones de pin en la interfaz de **Usuario del Anfitrión ECI**. A su vez, el **Usuario** puede seleccionar a través del **Anfitrión ECI** un **Cliente ECI** de entre los candidatos para realizar la autenticación parental utilizando la API de delegación de la autenticación parental (parAuthDel) definida en la cláusula 9.8.2.11. Esto puede ser de utilidad cuando un **Cliente ECI** que maneje numerosos elementos de contenido no pueda delegar su autenticación parental pero pueda realizar la autenticación parental en nombre del **Anfitrión ECI**.

Esta API también permite a un **Cliente ECI** iniciar una autenticación parental para un elemento de contenido antes de la apertura de una sesión de medios, por ejemplo, para la autenticación parental de un futuro evento de grabación.

9.8.2.10.2 Función de autenticación parental estándar

En esta cláusula se definen un conjunto de requisitos para una función de clasificación parental estándar basada en códigos pin de 4 caracteres que el **Anfitrión ECI** podrá ejecutar si lo solicita un **Cliente ECI** o que un **Cliente ECI** ejecutará en nombre del **Anfitrión ECI** si este ofrece ese servicio a través de la API de delegación de la autenticación parental.

Un **Anfitrión ECI** o un **Cliente ECI** pueden proporcionar una función de autenticación alternativa a la descrita a continuación en la presente cláusula si esa función proporciona al menos la integridad de autenticación parental del mecanismo definido en esta cláusula.

Las funcionalidades siguientes se aplican al mecanismo de autenticación parental basado en el código pin estándar:

1) La autenticación parental se basa en un código pin de al menos 4 caracteres alfanuméricos de entre un conjunto mínimo de 10 caracteres (por ejemplo, los dígitos).
2) La fijación del valor del valor del código pin estará protegido por el propio código pin o por un mecanismo de autenticación maestro que proteja el acceso a los activos o servicios de valor material cuyo acceso por menores se considera indeseable y a los que debe protegerse del contenido.
3) Cualquier valor límite de la clasificación parental aplicable estará protegido por el código pin o por un mecanismo de autenticación maestro como se señala en el párrafo anterior.
4) Los requisitos relativos a un mecanismo potencial de autenticación maestro crearán una integridad de autenticación que será como mínimo la del mecanismo de código pin definido en esta cláusula no basado en un mecanismo de autenticación maestro.
5) En la compra de un Anfitrión, el código pin inicial de clasificación parental o los medios para la autenticación solo se transferirá al propietario.
6) En la instalación de un nuevo cliente, el **Operador** sólo transferirá al propietario el código pin o los medios de autenticación mediante autenticación maestra.
7) El **Fabricante,** o un custodio que actúe en su nombre, puede proporcionar una forma de reiniciar el código pin con su valor inicial o proporcionar un servicio mediante el cual el propietario pueda fijar el código pin a un nuevo valor que sólo se transferirá al propietario.
8) El propio **Operador** puede proporcionar una forma de reasignar al código pin su valor inicial o proporcionar un servicio que permita al propietario fijar el código pin a un nuevo valor que sólo se transferirá al propietario.
9) En caso de 5 fallos consecutivos de autenticación en un periodo de 15 minutos, la función de autenticación parental rechazará cualquier nueva autenticación durante al menos 15 minutos.
10) No será posible recuperar o reiniciar el código pin mediante el software de **Usuario** ordinario, las aplicaciones descargadas ejecutadas en el **CPE** o en cualquier interfaz de **Usuario** o interfaces ordinarias.

9.8.2.10.3 Mensaje reqParAuthChk

C→H reqParAuthChk(ushort **mH) →**
C→H resParAuthChk(ushort **mH**, bool **ok)**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** que verifique la autenticación parental utilizando la función de autenticación parental estándar del **Anfitrión ECI** (véase la cláusula 9.8.2.10) y devolver el resultado en un mensaje de contestación.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del contenido a decodificar
ok: booleano	Es Verdadero si la autenticación tiene éxito; es Falso en cualquier otro caso, incluida una temporización.

Información de la Semántica:

- El **Anfitrión ECI** solo podrá tener una única verificación de autenticación parental pendiente por cada **Asa de Medios**. La emisión de una segunda petición para la misma **Asa de Medios** antes de responder o cancelar la anterior, generará dos **Contestaciones** idénticas.
- **reqParAuthChk**. El **Anfitrión ECI** debe arrancar una temporización cuando solicite una autenticación parental que finalizará transcurrido un tiempo razonable si no hay ninguna persona presente o en condiciones de realizar la autenticación tal como se propone en [b-UIT-T J Supl. 7].

9.8.2.10.4 Mensaje reqParAuthChkCan

C→H reqParAuthChkCan(ushort **mH) →**
H→C resParAuthChkCan(ushort **mH)**

- El **Cliente ECI** cancela cualquier petición de autenticación parental previa realizada a la **ECI**.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Postcondiciones a la Contestación:

1) El **Anfitrión ECI puede** devolver al **Cliente ECI** la contestación a un mensaje **reqParAuthChk** previo antes de que se produzca el mensaje **resParAuthChkCan**, pero no después.

9.8.2.10.5 Mensaje reqParAuthCid

H→C reqParAuthCid(uint **cidLength,** byte **cid**[]**) →**
C→H resParAuthCid(bool **ok)**

- Este mensaje permite a**l Anfitrión ECI** solicitar al **Cliente ECI** que realice cualquier autenticación necesaria para un futuro elemento de contenido identificado mediante el **cid**.

Definición de los parámetros de la Petición:

cidLength: uint	Longitud del parámetro **cid**.
cid[]: byte	Identificación del contenido sujeto a autenticación parental (si se requiere). El primer byte indica el formato del parámetro identificación de contenido tal como se define en el Cuadro 9.8.2.10.5-1.

Cuadro 9.8.2.10.5-1 – Formatos de identificación del contenido

Nombre	Valor	Descripción
CidDvbEvent	0x01	Identificación de evento DVB. Los bytes del **cid** siguen el orden siguiente: id de red original (2 bytes), id del flujo de transporte (2 bytes), id del servicio (2 bytes), id del evento (2 bytes) tal como se define en la tabla EIT en [ETSI EN 300 468]. Todos los campos de 2 bytes de la secuencia se representan con el orden de la red (con el byte más significativo en primer lugar).
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros de la Contestación:

ok: booleano	Verdadero si la autenticación parental ha sido exitosa o no es necesaria.

Información de la semántica

- El **Cliente ECI** mantendrá un registro no volátil de identificaciones de contenidos que han sido autenticadas con esta función. Si faltara espacio de almacenamiento se pueden descartar los registros más antiguos y los registros que han dejado tener utilidad. Los requisitos mínimos de esta memoria intermedia de identificación de contenido se proponen en [b-UIT-T J Supl. 7].

Los códigos de error conexos figuran en el Cuadro 9.8.2.10.5-2.

Cuadro 9.8.2.10.5-2 – Códigos de error de la API de sesión de medios para medios TS

Nombre	Valor	Descripción
ErrParAuthCidUnknOk	1	No ha sido posible identificar el estado de autenticación parental del elemento de contenido pero la autenticación parental se realizó correctamente.

Los estados de error anteriores también pueden devolverse en caso de no estar disponible el acceso a los recursos de red necesarios.

9.8.2.11 API de delegación de la autenticación parental

9.8.2.11.1 Generalidades

Esta API permite a un **Cliente ECI** indicar que puede realizar una función de autenticación parental estándar tal como se define en la cláusula 9.8.2.10.2 y al **Anfitrión ECI** delegar las verificaciones del código pin a dicho **Cliente ECI**.

Un **Cliente ECI** puede indicar que soporta la API de autenticación delegada utilizando la API de configuración en el momento de su inicialización.

NOTA – Al mismo tiempo, un **Cliente ECI** puede decidir no delegar su propia autenticación parental debido, por ejemplo, a consideraciones comerciales, de seguridad o jurídicas.

El **Anfitrión ECI** ofrecerá una función de configuración que permita al **Usuario** seleccionar al **Anfitrión ECI** para la autenticación del control parental estándar o para delegar la autenticación del control parental estándar a uno de los **Cliente ECI** que ofrece esa función.

9.8.2.11.2 Mensaje reqParAuthDel

H→C reqParAuthDel(ushort **mh**) →
C→H resParAuthDel(ushort **mH**, bool **ok)**

- Este mensaje permite al **Anfitrión ECI** solicitar al **Cliente ECI** que realice una autenticación parental delegada en su nombre para el contenido del mH.

Definición de los parámetros de la Petición:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del contenido a decodificar.
ok: booleano	Verdadero si la autenticación parental ha sido exitosa; falso si no lo ha sido o venció una temporización.

Información de la Semántica:

- El **Cliente ECI** solo podrá tener una verificación de autenticación parental pendiente por cada **Asa de Medios**. La emisión de una segunda petición para la misma **Asa de Medios** antes de responder a la anterior o cancelarla, generará dos contestaciones idénticas.
- El **Cliente ECI** debe arrancar una temporización cuando solicite una autenticación parental que vencerá en un plazo razonable si no hay nadie presente o en condiciones de realizar la autenticación tal como se propone en [b-UIT-T J Supl. 7].

9.8.2.11.3 Mensaje setParAuthDelCan

H→C reqParAuthDelCan(ushort **mH**) →
C→H resParAuthDelCan(ushort **mH)**

- Este mensaje permite al **Anfitrión ECI** cancelar una petición de autenticación parental delegada.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Definición de los parámetros de la Contestación:

mH: ushort	**Asa de Medios** del contenido a decodificar.

Postcondiciones a la Contestación:

- El **Anfitrión ECI** puede devolver al **Cliente ECI** la contestación a un mensaje reqParAuthDel previo antes de que se produzca el mensaje resParAuthDelCan, pero no después.

9.8.2.12 API de control del sistema de protección

9.8.2.12.1 Introducción

El contenido desencriptado por un **Cliente ECI** puede suministrarse por diferentes salidas del **CPE**. Las salidas suelen estar protegidas por un sistema de protección de salida. El sistema de protección de salida puede tener opciones para aceptar los mensajes de renovación del sistema (SRM) de un **Cliente ECI** y ofrecer al **Cliente ECI** la opción de bloquear las salidas a los dispositivos conectados a través del sistema de protección de salida en caso de que el ID de su dispositivo (en el contexto del sistema de protección de salida) figure en la lista como comprometida.

El sistema de protección puede gestionar varias salidas.

9.8.2.12.2 Mensaje getProtSystCtrl Message

C→H getProtSystCtrl()

- Este mensaje permite al **Cliente ECI** leer la lista de sistemas de protección de salida a los que da soporte el CPE, sus versiones y su soporte para SRM (Mensajes de capacidad de renovación del sistema) y los servicios de bloqueo de ID de dispositivos.

Cuadro 9.8.2.12.2-1 – Especificación del cuadro de control de protección

```
typedef struct ProtCtrlEl
em {
   ushort protSysType;    // tipo de sistema de protección según cuadro sect-2
   uint   srmSupp:4;      // nivel de soporte para SRMs según cuadro sect-3
   uint   devIdSupp:1;    // 0b0 no soporta para servicios de ID de
                          // dispositivo,
                          // 0b1 soporta servicios ID de dispositivo
   uint   reserved:11;    // reservado; tendrá valor 0b00000000000
} ProtCtrlElem;

#define MaxProtCtrlArr 32
typedef ProtCtrlElem ProtCtrlArr[MaxProtCtrlArr];
   // El sistema de protección enumerado en la matriz puede proteger múltiples
   // salidas.
   // Cada valor de ProtCtrlElem salvo cuando protSustType=0x0000 aparecerá
   // una sola vez en ProtCtrlArr. Todo ProtCtrlElem con ProtColElem distinto
   // de 0x0000
   // figurará en el elementos de índice más pequeño deProtCtrlArr,
   // los valores igual a 0x0000 figurarán al final de la matriz
```

Cuadro 9.8.2.12.2-2 – Valores para los tipos de sistemas de protección de salida

Nombre	Valor	Tipo de sistema de protección de salida
OpNoProtSyst	0x0000	Sin sistema de protección de salidas.
OpHDCP_1	0x0010	HDCP versión1.
OpHDCP_21	0x0011	HDCP versión 2.0 ó 2.1.
OpHDCP_22	0x0012	HDCP versión 2.2 o superior.
OpDTCP_1	0x0020	DCTP versión 1.
OpDTCP_2	0x0021	DTCP versión 2 o superior.
OpDTCP_IP1	0x0030	DTCP IP.
Proprietary	0x8xxx	Puede definirse fuera del ámbito de esta especificación.
Reserved	Other values	Reservado para uso futuro.

Cuadro 9.8.2.12.2-3 – Valores de soporte SRM

Protección	Valor	Tipo de sistema de protección de salida
SrmNone	0x0	Sin soporte SRM.
SrmProtSysSpecV1	0x1	Soporte de SRM con arreglo a la versión 1 (pero no superior) de la especificación del sistema de protección de salidas.
SrmProtSysSpecV2	0x2	Soporte de SRM con arreglo a la versión 2 (pero no superior) de la especificación del sistema de protección de salidas.
SrmProtSysSpecV3	0x3	Soporte de SRM con arreglo a la versión 3 (pero no superior) de la especificación del sistema de protección de salidas.
SrmProtSysSpecV4	0x4	Soporte de SRM con arreglo a la versión 4 (pero no superior) de la especificación del sistema de protección de salidas.
reserved	0x5..0xC	Reservado para uso futuro.
Proprietary	0xD-0xF	Puede definirse fuera del alcance de esta especificación.

Semántica:

- El soporte del servicio de ID de dispositivos significa que el sistema de protección deberá soportar la identificación y el bloqueo de cualquier conexión protegida a un dispositivo utilizando los mensajes reqBlockDevId, resBlockDevId.
- La configuración de las funciones de protección de salida será estática durante la "vida útil" del cliente.

9.8.2.12.3 Mensaje reqSrmMsg

C→H reqSrmMsg(ushort **protSysType**, uint **srmLen**, byte **srmData**[]**)** →
H→C resSrmMsg()

- Este mensaje permite al **Cliente ECI** enviar un SRM para el tipo de sistema de protección.

Definiciones de los parámetros de la petición:

protSysType[]: ushort	El tipo de sistema de protección al que apunta este SRM. Nota: Los SRM pueden aplicarse a múltiples tipos de la misma familia de sistemas de protección. En tal caso es suficiente enviar el SRM al anfitrión una sola vez y no para cada tipo.
srmLength: uint	Longitud del SRM.
srmData:byte[]	SRM.

Condiciones previas a la petición:

- No se envió ningún mensaje **reqSrmMsg** previo o se recibió el mensaje **resSrmMsg** al último mensaje **reqSrmMsg**.

Semántica detallada:

- El Anfitrión ECI enviará el menaje **resSrmMsg** lo antes posible.

Cuadro 9.8.2.12.3-1 – Códigos de error reqSrmMsg

Nombre	Descripción
ErrReqSrmMsgOverflow	Véase la cláusula 9.8.2.12.7.

9.8.2.12.4 Mensaje reqInfoDevId

H→C reqInfoDevId(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[]**)**→
C→H resInfoDevId(ushort **mh)**

- Este mensaje permite al **Anfitrión ECI** indicar los dispositivos (mediante **devId**) a los que se envía el contenido que el dispositivo puede desencriptar utilizando el sistema de protección **protSysType** en la sesión de desencriptado **mh**.

Definición de los parámetros de la petición:

mh: ushort	**Distintivo de medios** para la sesión de desencriptado en la que se utiliza el dispositivo con **devId**.
protSysType: ushort	Sistema de protección utilizado para proteger el contenido que se va a enviar al **devId** – véase el Cuadro 6.4.2-1 en [b-UIT-T J Supl. 7].
lenDevId: uint	Longitud del campo **devId** en bytes.
devId[]: byte	Device ID – codificación específica definida en una especificación complementaria.

Definición del parámetro de la respuesta:

mh: ushort	**Distintivo de medios** para la sesión de desencriptado para la que se proporciona respuesta.

Condiciones previas a la petición:

- No se envió ningún mensaje **reqInfoDevId** en la sesión **mh** o se recibió el mensaje **resInfoDevId** enviado al último mensaje **reqInfoDevId** en la sesión **mh**.

Semántica detallada:

- El **Anfitrión ECI** enviará el **devId** de cada dispositivo conectado a la salida de la sesión **mh** lo antes posible.

Cuadro 9.8.2.12.4-1 – Códigos de error reqInfoDevId

Nombre	Descripción
ErrReqInfoDevOverflow	Véase la cláusula 9.8.2.12.7.

9.8.2.12.5 Mensaje reqBlockDevId

C→H reqBlockDevId(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[]**)** →
H→C resBlockDevId(ushort **mh)**

- Este mensaje permite al **Cliente ECI** bloquear los dispositivos con **devId** a los que se envía el contenido desencriptado usando el sistema de protección **protSysType** en la sesión de desencriptación **mh**.

Definición de los parámetros de la petición:

mh: ushort	**Distintivo de medios** para la sesión de desencriptado en la que se utilizó el dispositivo con **devId**.
protSysType: ushort	Sistema de protección utilizado para proteger el contenido que se va a enviar al **devId** – véase el Cuadro 6.4.2-1 en [b-UIT-T J Supl. 7].
lenDevId: uint	Longitud del campo **devId** en bytes.
devId[]: byte	Device ID – La codificación específica se define en una especificación complementaria.

Definición de los parámetros de la contestación:

mh: ushort	**Distintivo de medios** para la sesión de desencriptado para la que se proporciona una contestación.

Condiciones previas a la petición:

- No se envió un **reqBlockDevId** en la sesión **mh** o se recibió el mensaje **resBlockDevId** para el último mensaje **reqBlockDevId** en la sesión **mh**.

Semántica:

- Al recibir un **reqBlockDevId** válido, el **Anfitrión ECI** responderá con **ErrReqOkNoId** (véase el Cuadro 9.3.4-1) y garantizará que la salida del dispositivo con **devId** quede bloqueada.

9.8.2.12.6 Mensaje setBlockProtSyst

C→H setBlockProtSyst(ushort **mh**, ushort **protSysType** bool **block)**

- Este mensaje permite al **cliente ECI** bloquear todo el contenido desencriptado utilizando el sistema de protección **protSysType** en la sesión de desencriptado **mh**.

Definición de parámetros:

mh: ushort	**Distintivo de medios** para la sesión de desencriptado en la que el contenido debe bloquearse.
protSysType: ushort	Sistema de protección para proteger el contenido que se ha de enviar a **devId** – véase el Cuadro 6.4.2-1 de [b-UIT-T J Supl. 7]
block: bool	**Verdadero** si el contenido se bloqueará, **Falso** en caso contrario.

Semántica:

- En caso de que **block** se modifique de **Verdadero** a **Falso** para un **protSysType** en una sesión **mh**, todos los **devID** para ese **protSysType** utilizados para la salida en la sesión **mh** los enviará el **Anfitrión ECI** utilizando el **reqInfoDevId** si la implementación de **protSysType** lo permite (como se indica en **getProtSystCtrl**).

9.8.2.12.7 Códigos de error para la API de control del sistema de protección

- En el Cuadro 9.8.2.12.7-1 se enumeran los códigos de error para la API de control del sistema de protección.

Cuadro 9.8.2.12.7-1 – Códigos de error relacionados con la API de control del sistema de protección

Nombre	Valor	Descripción
ErrReqSrmMsgOverflow	-256	El **Anfitrión ECI** indica que no puede aceptar todavía el próximo mensaje **next ReqSrmMsg**.
ErrReqInfoDevOverflow	-257	El **Cliente ECI** indica que todavía no puede aceptar el próximo mensaje **next ReqInfoDev**.

9.9 Conjunto de las API para la comunicación de Clientes y Aplicaciones ECI

9.9.1 Lista de las API definidas en esta cláusula

En el Cuadro 9.9.1-1 lista de las API incluidas en esta cláusula.

Cuadro 9.9.1-1 – API de recursos relacionados con la comunicación de clientes y aplicaciones ECI

Cláusula	Nombre de la API	Descripción
9.9.2	API para la comunicación entre clientes	Permite que un **Cliente ECI** establezca un trayecto de comunicación directo con otro **Cliente ECI**.

9.9.2 API para la comunicación entre clientes

9.9.2.1 Generalidades

El **Anfitrión ECI** ofrece un entorno para el intercambio normalizado de información entre **Clientes ECI** en forma de información de importación/exportación, URIs y contenido. Los **Clientes ECI** pueden comunicarse entre ellos para ofrecer una funcionalidad adicional (actualmente no definida en la **ECI**). Los **Clientes ECI** pueden registrar su capacidad más importante y predisposición para soportar la comunicación entre clientes mediante el recurso del descubrimiento (véase la cláusula 9.4.2). Tras la inicialización del sistema, pueden leer las identidades de otros **Clientes ECI,** incluidas las **Conexiones de importación/exportación** establecidas. Los **Clientes ECI** pueden abrir un canal de comunicación (denominado conducto) con un potencial par e intercambiar mensajes sobre ese conducto. El **Anfitrión ECI** cierra el conducto de un **Cliente ECI** cuando el **Cliente ECI** par detiene su actividad o se reinicializa.

El **Anfitrión ECI** ofrece al **Cliente ECI** identidades que se autentican mediante **Cadenas de certificados ECI** proporcionadas junto con los **Clientes ECI**. Los **Clientes ECI** proporcionarán un mecanismo de autenticación independiente adicional si la comunicación con un par pone en riesgo la seguridad.

En caso de comunicación entre un **Cliente ECI** que decodifica un contenido y otro **Cliente ECI** que posteriormente reencripta ese contenido (un **Microservidor**), la recomendación para el establecimiento de un conducto es que este sea iniciado (abierto) por el **Microservidor**.

En el Cuadro 9.9.2.1-1 se muestran los mensajes de la API para la comunicación entre clientes.

Cuadro 9.9.2.1-1 – Mensajes de la API para la comunicación entre clientes

Mensaje	Tipo	Dir.	Etiqueta	Descripción
getIccMaxClients	S	C→H	0x0	El **Cliente ECI** lee el número máximo de **Clientes ECI** que puede soportar el **Anfitrión ECI**.
reqIccSystemReady	A	H→C	0x1	El **Anfitrión ECI** informa al **Cliente ECI** que todos los **Clientes ECI** se han inicializado.
getIccClientInfo	S	C→H	0x2	El **Cliente ECI** lee la identidad y el estado de la conexión de otro **Cliente ECI** en el sistema.
reqIccPipeOpen	A	C→H	0x3	Solicita la apertura de un conducto con otro **Cliente ECI**.
reqIccPipeOpenReq	A	H→C	0x4	Petición entrante de otro **Cliente ECI** para abrir un conducto.
reqIccPipeCancel	A	C→H	0x5	El **Cliente ECI** cancela el conducto.
reqIccPipeClose	A	H→C	0x6	El **Anfitrión ECI** informa al **Cliente ECI** que el conducto con el par ha sido cerrado.
reqIccPipeMsgSend	A	C→H	0x7	El **Cliente ECI** envía un mensaje a su par en un conducto.
reqIccPipeMsgRecv	A	H→C	0x8	El **Cliente ECI** recibe un mensaje de su par en un conducto.

9.9.2.2 Mensaje getIccMaxClients

C→H uint getIccMaxClients()

- Obtiene el número máximo de **Clientes ECI** que puede soportar el **Anfitrión ECI**.

Definición de propiedades:

- Entero sin signo que representa el número máximo de **Clientes ECI** que puede soportar el **Anfitrión ECI**.

9.9.2.3 Mensaje reqIccSystemReady

H→C reqIccSystemReady()

- El **Anfitrión ECI** informa al **Cliente ECI** que se han inicializado todos los demás **Clientes ECI**.

Semántica:

- Este mensaje se proporciona durante la inicialización del sistema para indicar a todos los **Clientes ECI** registrados en esta API que es posible comenzar a leer el registro de información de clientes e intentar abrir conductos con otros **Clientes ECI**.
- El campo ConnId del resultado refleja el último estado conocido de las **Conexiones de importación/exportación** del **Cliente ECI** con un potencial par. Están sujetas a posibles cambios.
- No se requiere un mensaje del resultado.

9.9.2.4 Mensaje getIccClientInfo

C→H ClientInfo getIccClientInfo(ushort **clientId)**

- El **Cliente ECI** lee la identidad y situación de conexión de otro **Cliente ECI** en el sistema.

Definición de los parámetros:

clientId: ushort	Id del cliente para el establecimiento de conductos. Este identificador no cambia a lo largo del ciclo de vida del sistema. Se modifica durante la reinicialización.

Definición de propiedades:

- connectionID es una propiedad dinámica.

- ClientInfo es una estructura que proporciona la identidad del **Cliente ECI** designado y cualquier **Conexión de importación/exportación** con ese **Cliente ECI**. Se define a continuación.

Definición de tipos de 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];
}
```

Definición de campos:

operatorId: ECI_Operator_Id	ID de operador del **Cliente ECI**.
platformOperationId: ECI_Platform_Operation_Id	ID de operación de plataforma del **Cliente ECI**.
client: union	Puede ser ECI_Client_Series_Id o ECI_Client_Id. El campo tipo de clientSeriesId y de clientId determina si es un clientSeriesId o un clientId.
VendorId: ECI_Vendor_Id	ID de suministrador del **Cliente ECI**.
clientSeriesId: ECI_Client_Sesies_Id	ID de series de clientes del **Cliente ECI**.
clientId: ECI_Client_Id	ID de cliente del **Cliente ECI**.
connId: ushort[MaxConnId]	Matriz de identificadores de conexión; el valor 0xFFFF señaliza una entrada vacía de la matriz. Todas las entradas vacías de la matriz se encuentran al final de la misma.

9.9.2.5 Mensaje reqIccPipeOpen

C→H reqIccPipeOpen(ushort **clientId**, byte **protocolId[16]) →**
H→C resIccPipeOpen(ushort **clientId)**

- Este mensaje permite al **Cliente ECI** solicitar al **Anfitrión ECI** la apertura de un conducto con otro **Cliente ECI**.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente al que se solicita un conducto.
protocolId[16]: byte	ID del protocolo de mensajes a utilizar. Será un UUID [IETF RFC 4122] cuyos octetos mantienen el orden de red en la matriz.

Definición de los parámetros del resultado

clientId: ushort	ID del cliente al que se solicitó la apertura de un conducto.

Precondiciones a la Contestación:

- El conducto está abierto o se devuelve un código de error. Los códigos de error conexos figuran en el Cuadro 9.9.2.5-1.

Cuadro 9.9.2.5-1 Códigos de error de reqIccPipeOpen

Nombre	Descripción
ErrIccPipeOpenReject	Véase el Cuadro 9.9.2.11-1.
ErrIccPipeOpenNoConn	
ErrIccPipeOpenProtocol	
ErrIccPipeOpenNotReady	

9.9.2.6 Mensaje reqIccPipeOpenReq

H→C reqIccPipeOpenReq(ushort **clientId**, byte **protocolId**[16]**) →**
C→H resIccPipeOpen (ushort **clientId)**

- Este mensaje permite al **Cliente ECI** recibir una petición entrante de otro **Cliente ECI** para la apertura de un conducto a través del **Anfitrión ECI**.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente que solicita un conducto.
protocolId[16]: byte	ID del protocolo de mensajes a utilizar. Será un UUID [IETF RFC 4122] cuyos bytes mantienen el orden de la red.

Definición de los parámetros del resultado:

clientId: ushort	ID del cliente que solicitó el conducto.

Semántica:

- El valor de la contestación de clientId será idéntica al valor de la petición.

Precondiciones a la Contestación:

- El **Cliente ECI** puede rechazar el conducto. Los códigos de error son los mismos que los de la apertura de un conducto y se transportan de forma transparente al peticionario. Figuran en el Cuadro 9.9.2.5-1.

9.9.2.7 Mensaje reqIccPipeCancel

C→H reqIccPipeCancel(ushort **clientId) →**
H→C resIccPipeCancel(ushort **clientId)**

- Este mensaje permite al **Cliente ECI** indicar al **Anfitrión ECI** que desea dar por terminado el conducto.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente del conducto que se desea cancelar.

Definición de los parámetros del resultado:

clientId: ushort	ID del cliente del conducto cancelado.

Semántica:

- El valor de la contestación del clientId será idéntica al valor de la petición.

Precondición a la Contestación:

- El conducto se da por terminado: el **Cliente ECI** que solicita la cancelación del conducto no recibirá ningún otro mensaje a través del conducto.

Información de la Semántica:

- Si el conducto no estaba abierto no se produce ningún error.

9.9.2.8 Mensaje reqIccPipeClose

H→C reqIccPipeClose(ushort **clientId**, uint **reason)** →
C→H resIccPipeClose(ushort **clientId)**

- Este mensaje permite al **Anfitrión ECI** informar al **Cliente ECI** de que el conducto con el par se ha cerrado.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente del conducto que se ha cerrado.
reason: uint	Motivo del cierre del conducto. Los valores figuran en el Cuadro 9.9.2.11-1.

Cuadro 9.9.2.8-1 – Valores asociados al motivo de reqIccPipeClose

Nombre	Valor	Descripción
IccPipeCloseCancel	0x01	Conducto cerrado por un par mediante un mensaje reqIccPipeCancel.
IccPipeCloseStop	0x02	Conducto cerrado por el **Anfitrión ECI** como consecuencia de la terminación por el **Cliente ECI** par. Es posible que a continuación se reinicialice el **Cliente ECI**.
RFU	Otros	Reservado para uso futuro.

Definición de los parámetros del resultado:

clientId: ushort	ID del cliente del conducto que se ha cerrado.

Precondición a la Petición:

- No se enviarán más mensaje a través del conducto.

Precondición a la Contestación:

- El **Cliente ECI** no intentará enviar nuevos mensajes a través del conducto (cerrado).

9.9.2.9 Mensaje reqIccPipeMsgSend

C→H reqIccPipeMsgSend(ushort **clientId**, uint **msgId**, uint **dataLen**, byte **data[])** →
H→C resIccPipeMsgSend(ushort **clientId)**

- Este mensaje permite al **Cliente ECI** enviar un mensaje a su par de un conducto. Los códigos de error conexos figuran en el Cuadro 9.9.2.11-1.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente al que se ha enviado el mensaje.
msgId: uint	Id del mensaje. Todos los valores negativos y cero están reservados; todos los valores positivos son específicos de la aplicación (su significado se define en el contexto del emisor y receptor).
dataLen: uint	Longitud en número de bytes del parámetro datos. No excederá de 32 768.
data[]: byte	Campo de datos para el mensaje.

Definición de los parámetros del resultado:

clientId: ushort	ID del cliente del conducto.

Precondición a la Petición:

- El siguiente mensaje reqIccMsgSend solo puede enviarse una vez recibido el mensaje resIccMsgSend anterior por el mismo conducto.

Cuadro 9.9.2.9-1 – Códigos de error de reqIccPipeMsgSend

Nombre	Descripción
ErrIccPipeClosed	Véase el Cuadro 9.9.2.11-1.

9.9.2.10 Mensaje reqIccPipeMsgRecv

H→C reqIccPipeMsgRecv(ushort **clientId**, uint **msgId**, uint **dataLen**, byte **data[]) →**
C→H resIccPipeMsgRecv(ushort **clientId)**

- Este mensaje permite al **Cliente ECI** recibir un mensaje de su par en un conducto.

Definición de los parámetros de la Petición:

clientId: ushort	ID del cliente del que se recibe el mensaje.
msgId: uint	Id del mensaje. Todos los valores negativos y cero están reservados; todos los valores positivos son específicos de la aplicación (su significado se define en el contexto del emisor y receptor).
dataLen: uint	Longitud en número de bytes del parámetro datos. No excederá de 32 768.
data: byte[]	Campo de datos para el mensaje.

Definición de los parámetros del resultado:

clientId: ushort	ID del cliente del conducto.

Precondición a la Petición:

- El siguiente mensaje reqIccMsgRecv solo se enviará tras haber recibido el mensaje resIccMsgRecv anterior por el mismo conducto.

9.9.2.11 Códigos de error para las comunicaciones entre clientes

Los códigos de error de las API para la comunicación entre clientes figuran en el Cuadro 9.9.2.11-1.

Cuadro 9.9.2.11-1 – Códigos de error de las comunicaciones entre clientes

Nombre	Valor	Descripción
ErrIccPipeOpenReject	-256	El par ha rechazado el conducto.
ErrIccPipeOpenNoConn	-257	El par rechaza el conducto como consecuencia de no existir una **Conexión de importación/exportación** con el Cliente ECI.
ErrIccPipeOpenProtocol	-258	El par rechaza el protocolo propuesto para el conducto.
ErrIccPipeOpenNotReady	-259	El par no está en un estado que le permita estar preparado para aceptar un conducto. Es adecuado reintentar posteriormente el establecimiento de un conducto.
ErrIccPipeClosed	-260	El conducto está cerrado.

10 Funcionalidades obligatorias y opcionales del Anfitrión ECI

10.1 Introducción

Las especificaciones técnicas del sistema **ECI** soportan soluciones técnicas para una amplia gama de **CPE** adecuados para el consumo de medios. Es decisión del **fabricante del CPE** determinar las funciones frontales, básicas y de soporte que implementa en su dispositivo. En el caso de las funcionalidades frontales y de soporte, probablemente el fabricante sólo implementará las API de la **ECI** que mejor se adapten a su hardware/pila de protocolos. A fin de dotar de flexibilidad al **Usuario**, en el Cuadro 10.2-1 se enumeran todas las API, tanto obligatorias (m, *mandatory*), como opcionales (o) y condicionales (c) para las distintas categorías de **CPE**.

10.2 Lista de funcionalidades ECI obligatorias y opcionales para distintos tipos de dispositivos CPE

En el Cuadro 10.2-1 figuran las funcionalidades **ECI** obligatorias y opcionales para distintos tipos de dispositivos **CPE**. La implementación de las diversas API depende de la disponibilidad de determinados componentes hardware/software en el dispositivo **CPE**.

Cuadro 10.2-1 – Lista de funcionalidades ECI obligatorias y opcionales

API	Cláusula	Anfitrión	Condición (si procede)	Cliente de desencrip.	Micro servidor	Micro cliente
Descubrimiento de la interfaz del anfitrión	9.4.2	M		M	M	M
MMI	9.4.3	M		O	O	O
IP	9.4.4	C	Si soporta conectividad IP	O	O	O
HTTP(S)	9.4.4.6	M		O	O	O
Sistema de ficheros	9.4.5	M		O	O	O
Temporizador y reloj	9.4.6	M		O	O	O
Gestión de la energía	9.4.7	M		O	O	O
Fijación de país e idioma	9.4.8	M		O	O	O
Seguridad avanzada general	9.5.2.2	M		M	M	M
Desencriptación de seguridad avanzada	9.5.2.3	M		M	N.A.	M
Exportación de seguridad avanzada	9.5.2.4	C	Para registro o pasarela	O	N.A.	O
Encriptación de seguridad avanzada	9.5.2.5	C	Para registro o pasarela	N.A.	M	N.A.
Tarjeta inteligente	9.5.3	C	Para lector SC soportado	O	O	O
Carrusel de datos	9.5.4	C	Para red de difusión	O	O	O
Desencriptación (véase la nota)	9.6.2	M		M	N.A.	M
Conexión de exportación	9.7.2.3	C	Para registro o pasarela	O	N.A.	O
Conexión de importación	9.7.2.4	C	Para registro o pasarela	N.A.	M	N.A.
Reencriptación (véase nota)	9.7.2.5	C	Para registro o pasarela	N.A.	M	N.A.
Desencriptación de Microcliente	9.7.2.6	M		O	N.A.	M
Fijación de país e idioma	9.4.8	M		O	O	O
URI estándar	9.8.2.3	M		M	M	M
URI de cliente	9.8.2.4	M		M	M	M
URI básica	9.8.2.5	M		M	M	M
Control de salida	9.8.2.6	M		M	M	M
Filigrana	9.8.2.7	O		O	N.A.	O
Control parental	9.8.2.8	M		M/O	M/O	M/O
Sincronismo de propiedades del contenido	9.8.2.9	M		M	M	M
Autenticación parental	9.8.2.10	M		O	N.A.	O
Delegación de autenticación parental	9.8.2.11	M		O	N.A.	O
Comunicación entre clientes	9.9.2	M		O	O	O
NOTA – Pueden designarse intervalos específicamente para **Microservidores** y para clientes de la desencriptación. El intervalo es técnicamente idéntico, pero los recursos de **AS** necesarios y las funciones de desaleatorización asociadas son diferentes.						

La API de descubrimiento no ofrece un mecanismo que permita a un **Anfitrión ECI** detectar que un **Cliente ECI** puede desencriptar o encriptar datos de medios con formato fichero y/o de flujo de transporte. El campo mhType del parámetro decryptId del mensaje setDcrMhMatch proporciona la señalización necesaria (véase la cláusula 9.6.2.2.2). El parámetro EciEncrModes del mensaje setEncrModes proporciona ese descubrimiento con fines de reencriptación (véase la cláusula 9.7.2.5.3).

- Un dispositivo conforme con **ECI** sólo a efectos del consumo, proporcionará al menos 2 instancias de VM e intervalos de SA.
- Los **Anfitriones ECI** que soporten la funcionalidad PVR admitirán que exista al menos un contenedor adicional (instancia de **VM**) y un **intervalo-AS** para un **Microservidor**. Si esos **Anfitriones ECI** también proporcionan la funcionalidad de reproducción del contenido almacenado, soportarán al menos un contenedor adicional (instancia de VM) y un intervalo-**AS** para un **Microcliente** que pueda decodificar el contenido reencriptado.
- Los **Anfitriones ECI** que soportan la funcionalidad de pasarela en red permitirán que exista al menos un contenedor adicional (instancia de **VM**) e intervalo **AS** para un **Microservidor**.

Anexo A

Funciones criptográficas del Anfitrión ECI

(Este anexo es parte integrante de esta Recomendación.)

A.1 Función hash

Todas las funciones *hash* (troceo y cifrado) utilizadas en la presente Recomendación se basan en el algoritmo SHA256 definido en [NIST FIPS 197].

La función *hash* de la cláusula 5.2 es SHA-256() tal como se define en [NIST FIPS 197].

La función-c asHash(uchar *data, uint datalength, resultLength, uchar *result) utiliza los octetos comenzando con datos de longitud dataLength como cadenas de octetos *dataIn* y calcula la cadena de octetos resultOut como cadena de octetos resultLength/8, que almacena en el campo de resultado conforme a lo siguiente:

resultOut = BS2OSP(truncate(SHA-256(OS2BSP(*dataIn*)),resultLength)))

resultLength es un múltiplo de 8. 'Truncate' es la función que trunca a izquierdas una cadena de bits (parámetro 1) a una longitud máxima de (parámetro 2) bits.

BS2OSP y OS2BSP son funciones que convierten una cadena de bits en una cadena de octetos y viceversa, tal como se define en la cláusula 9 de [UIT-T J.1014].

A.2 Criptografía asimétrica

Las operaciones de encriptación y desencriptación asimétricas se definen en la cláusula 12.4 de [UIT-T J.1014].

A.3 Criptografía simétrica

La criptografía AES de esta Recomendación es la definida en [NIST FIPS 197] salvo que se proporcione una referencia de aplicación específica para una aplicación AES.

Las aplicaciones CBC de AES serán como se define en [NIST Block 2001], salvo que se proporcione una referencia de aplicación específica para CBC con AES. Si no se define otra cosa, se utilizará el vector 0 de inicialización.

Las aplicaciones CTR de AES serán como se define en [NIST Block 2001] salvo que se proporcione una referencia de aplicación específica para CTR con AES. Si no se define otra cosa, se utilizará el vector 0 de inicialización.

A.4 Generación de números aleatorios

La generación de números aleatorios definida en la presente Recomendación será conforme a lo especificado en el Anexo A de [UIT-T J.1014].

Anexo B

Parámetros de interoperabilidad

(Este anexo es parte integrante de esta Recomendación.)

B.1 Introducción

En este Anexo se definen parámetros relacionados con requisitos asociados a los recursos de los **CPE**. El cumplimiento de estos requisitos permite la interoperabilidad entre los **Clientes ECI**, los servicios de seguridad **ECI** prestados por las redes y los **CPE**.

B.2 Longitud de las listas de revocación

Los **CPE** reservarán un espacio de almacenamiento **NV** suficiente para **Listas de revocación** (RL) de longitudes recogidas en el Cuadro B.2-1 para cada elemento que pueda ser revocado. La **TA ECI** debe garantizar que las RL de la **TA ECI** generadas respeten esos límites.

Cuadro B.2-1 – Longitud máxima de las Listas de Revocación

Lista de Revocación	Número máximo de identificadores
RL de fabricante	500
RL de anfitrión	500
RL de suministrador	500
RL de Cliente ECI	500
RL de operador	500
RL de Operación de Plataforma	500

B.3 Tamaño de la imagen del cliente ECI

Un **Anfitrión ECI** dispondrá de una capacidad de almacenamiento mínima para la **Imagen de Cliente ECI** de 500 Kbytes por cada intervalo de **Cliente ECI** que soporte.

B.4 Parámetros de la configuración del carrusel de difusión

La **ECI** define tiempos de adquisición máximos tCdownloadScenario para todos los elementos que se descargan de un carrusel de difusión a fin de permitir un diseño adecuado de los **Anfitriones ECI**. El parámetro tCdownloadScenario refleja el tiempo real de descarga; por lo tanto, la tasa de repetición del carrusel debería ser al menos un múltiplo del triple de dicho parámetro fin de garantizar la descarga por el **Anfitrión ECI** dentro de estos límites. Los difusores deberían proporcionar una anchura de banda adecuada para soportar la tasa de repetición requerida.

La **ECI** también define un tamaño de módulo máximo para la atribución de memoria intermedia.

En el Cuadro B.4-1 se definen el tCdownloadScenario y el tamaño máximo del módulo para cuya gestión debería estar diseñado el **Anfitrión ECI**.

Cuadro B.4-1 – Periodos máximos del escenario de descarga y tamaños del módulo de carruseles ECI

Tipo de tabla	tCdownloadScenario	Tamaño máximo del módulo
Imágenes de Cliente ECI	5 minutos	500 Kbyte
Datos de revocación **de Cliente ECI**	5 minutos	100 Kbyte por colector
Cadena de certificados de Operación de Plataforma	10 segundos	50 Kbyte
Datos de revocación de **Operación de Plataforma**	5 minutos	100 Kbyte por colector
Datos de revocación de **Anfitrión ECI**	5 minutos	100 Kbyte por colector
Datos de configuración de **AS**	2 minutos	20 Kbyte por colector

Anexo C

Visión general de las API del Anfitrión ECI

(Este anexo es parte integrante de esta Recomendación.)

En el Cuadro C-1 se definen los valores de **MsgApiTag** con arreglo a la cláusula 9.3.1.

Cuadro C-1 – Esquema de numeración de las API de la ECI

API	Cláusula	Valor de MsgApiTag	Versión más alta de la API	Versiones obsoletas de la API
Descubrimiento de la interfaz del anfitrión	9.4.2	0x0001	0x0000	ninguna
MMI	9.4.3	0x0002	0x0000	ninguna
IP	9.4.4	0x0003	0x0000	ninguna
HTTP(S)	9.4.4.6	0x0004	0x0000	ninguna
Sistema de ficheros	9.4.5	0x0005	0x0000	ninguna
Temporizador y reloj	9.4.6	0x0006	0x0000	ninguna
Gestión de la energía	9.4.7	0x0007	0x0000	ninguna
Fijación de país e idioma	9.4.8	0x0008	0x0000	ninguna
Seguridad avanzada general	9.5.2.2	0x0009	0x0000	ninguna
Desencriptación de seguridad avanzada	9.5.2.3	0x000A	0x0000	ninguna
Exportación de seguridad avanzada	9.5.2.4	0x000B	0x0000	ninguna
Encriptación de seguridad avanzada	9.5.2.5	0x000C	0x0000	ninguna
Tarjeta inteligente	9.5.3	0x000D	0x0000	ninguna
Carrusel de datos	9.5.4	0x000E	0x0000	ninguna
Desencriptación	9.6.2	0x000F	0x0000	ninguna
Conexión de exportación	9.7.2.3	0x0010	0x0000	ninguna
Conexión de importación	9.7.2.4	0x0011	0x0000	ninguna
Reencriptación	9.7.2.5	0x0012	0x0000	ninguna
Desencriptación de Microcliente	9.7.2.6	0x0013	0x0000	ninguna
URI estándar	9.8.2.3	0x0014	0x0000	ninguna
URI de cliente	9.8.2.4	0x0015	0x0000	ninguna
URI básica	9.8.2.5	0x0016	0x0000	ninguna
Control de salida	9.8.2.6	0x0017	0x0000	ninguna
Marca de agua	9.8.2.7	0x0018	0x0000	ninguna
Control parental	9.8.2.8	0x0019	0x0000	ninguna
Sincronismo de propiedades del contenido	9.8.2.9	0x0020	0x0000	ninguna
Autenticación parental	9.8.2.10	0x0021	0x0000	ninguna
Delegación de la autenticación parental	9.8.2.11	0x0022	0x0000	ninguna
Comunicación entre clientes	9.9.2	0x0023	0x0000	ninguna

Anexo D

Compatibilidad hacia adelante de las definiciones de propiedades del contenido

(Este anexo es parte integrante de esta Recomendación.)

Las propiedades del contenido deben implementarse de forma muy robusta utilizando hardware o firmware de bajo nivel, pudiendo resultar complejo, costoso o imposible su modificación o mejora tras la producción del SOC. En la presente cláusula se explica el enfoque para la evolución de dichas propiedades del contenido pese a las limitaciones de esa mejora.

En el futuro podrán necesitarse nuevas propiedades del contenido y/o alguna funcionalidad ampliada de las propiedades del contenido existentes. Ello puede incluir la ampliación del número de bits que representan el valor de una propiedad del contenido. La implementación de las propiedades del contenido en un **Anfitrión ECI** más antiguo desconoce nuevas funcionalidades y a menudo no es posible actualizarla. La definición de propiedades del contenido en **Anfitriones ECI** es tal que se consigue la máxima compatibilidad hacia adelante con respecto a las nuevas funcionalidades de las propiedades del contenido.

Los **Anfitriones ECI** tendrán un comportamiento definido para todos los valores de entrada e ignorarán cualquier ampliación de campo para la que no estén diseñados. Asimismo, tendrán un comportamiento inequívoco, es decir, cada valor de una futura propiedad de contenido tendrá un *único comportamiento definido* para todos los **Anfitriones ECI** que no implementen todas las ampliaciones, incluidos los **Anfitriones ECI** que cumplan la primera versión de propiedades del contenido. Al aplicar este principio, pueden asignarse nuevos valores de propiedades de contenido con pleno conocimiento de los efectos que ello tendrá sobre el comportamiento de implementaciones anteriores del **Anfitrión ECI**. Si una nueva propiedad de contenido tuviera dos (o más) opciones distintas de interpretación debidas a la interpretación de la retrocompatibilidad por parte de los **Anfitriones ECI** más antiguos, pueden asignarse dos (o más) valores reservados con la misma semántica de propiedades del contenido en la definición de las nuevas propiedades del contenido, cada una con una interpretación retrocompatible adecuada (pero distinta).

Un ejemplo de ampliación de un campo es la definición un nuevo campo control de salidas para un nuevo tipo de salida X en la API de control de salidas. Se asigna al bit 5, que está reservado en la versión 1. Puede utilizar la equivalencia semántica del campo OcIP. Cualquier implementación anterior de **Clientes ECI** asignará 0 a este campo. La interpretación que hará un **Anfitrión ECI** más antiguo será la siguiente:

- Si OcAnyOther==0b0 se permite OutputX.
- Si OcAnyOther==0b1 no se permite OutputX.

Ello corresponde por completo a la semántica de una nueva implementación de **Anfitrión ECI** cuando OcX==0b0. No obstante, cuando OcX==0b1 el permiso de salida será el inverso a la configuración anterior con OcX== 0b0, permitiendo una nueva funcionalidad en la combinación de un nuevo **Anfitrión ECI** y un nuevo **Cliente ECI**. Obsérvese que la interpretación inversa de los valores de campos que dependen de OcAnyOther garantiza que el valor 0 de cualquier campo no definido adopte su significado natural: permiso máximo para OcAnyOther==0b0 (otras salidas están permitidas) y permiso mínimo para OcAnyOther= 0b1 (otras salidas no están permitidas).

A la viceversa, es importante que los **Clientes ECI** que no utilicen la definición más reciente de las propiedades del contenido no apliquen de forma inadvertida nuevas funcionalidades de definiciones de propiedades del contenido posteriores que desconozcan, o aún peor, que utilicen dichos valores, presumiblemente no asignados, con fines privados basado en el hecho de que esos valores tienen un comportamiento predeterminado en todos los **Anfitriones ECI**. Normalmente, ese uso inadecuado creará un grave obstáculo para la futura incorporación de esos valores para fines definidos en la **ECI**. Por lo tanto, esta especificación prohíbe explícitamente la aplicación por los **Clientes ECI** de valores de propiedades del contenido no asignadas.

Específicamente, para campos con varios valores, todos los valores reservados tendrán un comportamiento definido en los **Anfitriones ECI**, pero los **Clientes ECI** no utilizarán los valores reservados.

Cualquier subcampo no asignado en una definición de propiedades del contenido tendrá un determinado comportamiento en un **Anfitrión ECI**, que corresponde a uno de los valores definidos de propiedades del contenido. Normalmente, un **Anfitrión ECI** ignorará esos subcampos, es decir, el **Anfitrión ECI** interpreta el valor de las propiedades del contenido simplemente con arreglo a los campos que han sido definidos. Normalmente, los **Clientes ECI** asignarán el valor 0 a esos subcampos. Cualquier desviación de la política correspondiente a un valor de subcampo no asignado igual a cero se predefinirá mediante una versión de la definición de propiedades del contenido.

Los **Anfitriones ECI** conformes con la correspondiente definición de propiedades del contenido ignorarán las ampliaciones de campos y los **Clientes ECI** que asignen valores asignarán el valor 0 a dichas ampliaciones de campo.

Apéndice I

Lista de todos los mensajes API disponibles en orden alfabético

(Este anexo no es parte integrante de esta Recomendación.)

Los mensajes API que figuran en el Apéndice I están extraídos de los cuadros de la cláusula 9 de la presente Recomendación que figuran en el Cuadro I-1.

Cuadro I-1 – Lista de Cuadros donde se presentan los mensajes de las distintas API

API	Cuadro	Categoría de API
API de descubrimiento de la interfaz del Anfitrión	9.4.2.1-1	API generales
API de la interfaz del usuario	9.4.3.1-1	
API de conectores IP	9.4.4.3.1-1	
API de conectores UDP	9.4.4.4.1-1	
API de conectores TDP	9.4.4.5.1-1	
API Get HTTP	9.4.4.6.1-1	
API de apertura / cierre de ficheros	9.4.5.2.1-1	
API de acceso a ficheros	9.4.5.3.1-1	
API del servicio de directorio de ficheros	9.4.5.4.1-1	
API del temporizador	9.4.6.2.1-1	
API del reloj	9.4.6.3.1-1	
API de transición energética	9.4.7.2-1	
API para despertar del estado de reposo	9.4.7.3-1	
API de establecimiento de país/idioma	9.4.8.1-1	
API de seguridad avanzada general	9.5.2.2.1-1	API específicas de la ECI
API de desencriptación de seguridad avanzada	9.5.2.3.1-1	
API de exportación de seguridad avanzada	9.5.2.4.1-1	
API de encriptación de seguridad avanzada	9.5.2.5.1-1	
API de gestión de sesión de **Tarjeta inteligente**	9.5.3.6.1-1	
API de comunicación de **Tarjeta inteligente**	9.5.3.6.1-1	
API de adquisición del carrusel de datos	9.5.4.1-1	
API de sesión de desencriptación de distintivo de medios	9.6.2.2.1-1	
API de conexión de exportación	9.7.2.3.1-1	
API de conexión de importación	9.7.2.4.1-1	
API de reencriptación	9.7.2.5.1-1	
API de desencriptación	9.7.2.6.1-1	
API de derechos de uso y control parental	9.8.2.1-1	
API para la comunicación entre clientes	9.9.2.1-1	

En el Cuadro I-2 se enumeran todos los mensajes de las API en orden alfabético.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
1	callAsNextKeySession	Seguridad avanzada general	9.5.2.2.3	S	C→H	Cambio a la siguiente clave aleatoria para una sesión.
2	callCardGetProp	**Tarjeta inteligente**	9.5.3.6.5	S	H→C	Obtiene las propiedades/parámetros de las comunicaciones mediante tarjeta.
3	callCardSessionPrio	**Tarjeta inteligente**	9.5.3.5.3	S	C→H	Fija la prioridad de la sesión de **Tarjeta inteligente**.
4	callCardSetProp	**Tarjeta inteligente**	9.5.3.6.4	S	H→C	Fija los parámetros de comunicación de la tarjeta.
5	callFileDataLog	Sistema de ficheros	9.4.5.3.6	S	C→H	Añade datos al final de un fichero almacenado en memoria intermedia.
6	callLocaltime	Reloj	9.4.6.3.3	S	C→H	Convierte un valor entero de tiempo en la hora local.
7	getApis	Descubrimiento de la interfaz	9.4.2.2	S	C→H	Obtiene las API de anfitrión disponibles.
8	getApiVersions	Descubrimiento de la interfaz	9.4.2.3	S	C→H	Obtiene las versiones disponibles de una API de anfitrión.
9	getAsClientRnd	Seguridad avanzada general	9.5.2.2.13	S	C→H	Obtiene un nuevo número aleatorio para las aplicaciones de **Cliente ECI**.
10	getAsSC	Seguridad avanzada general	9.5.2.2.14	S	C→H	Obtiene el estado actual del campo control de aleatorización del contenido de una sesión
11	getAsSessionLimitCounter	Seguridad avanzada general	9.5.2.2.10	S	C→H	Obtiene el valor límite actual del contador de la sesión.
12	getAsSessionRk	Seguridad avanzada general	9.5.2.2.9	S	C→H	Obtiene el valor aleatorio de la clave para una sesión.
13	getAsSlotRk	Seguridad avanzada general	9.5.2.2.8	S	C→H	Obtiene el valor aleatorio de la clave para el **intervalo SA.**
14	getCardConnStatus	Tarjeta inteligente	9.5.3.5.4	S	H→C	Proporciona el estado de conexión de la tarjeta.
15	getChipsetId	Seguridad avanzada general	9.5.2.2.16	S	H→C	Obtiene el valor de ChipsetID del **bloque de escalera de claves**
16	getDcrMarkMeta	Propiedades del contenido	9.8.2.7.4	S	H→C	Lee una propiedad del sistema de marcaje
17	getDcrMarkSyst	Propiedades del contenido	9.8.2.7.2	S	H→C	Obtiene los sistemas de marcaje permitidos.
18	getDcrTsSource	Control de la fuente del TS para desencriptación	9.6.2.3.6.2	S	C→H	El **Cliente ECI** obtiene la fuente del TS.
19	getEncrStdUri	Propiedades del contenido	9.8.2.3.2	S	C→H	Obtiene la URI estándar del contenido a reencriptar.
20	getEncrBasicUri	Propiedades del contenido	9.8.2.5.2	S	C→H	Obtiene la URI básica del contenido a reencriptar
21	getEncrCustUri	Propiedades del contenido	9.8.2.4.2	S	C→H	Obtiene la URI a medida del contenido a reencriptar
22	getEncrOutputCtrl	Propiedades del contenido	9.8.2.6.2	S	C→H	Obtiene las restricciones del control de salida para el contenido a reencriptar.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
23	getEncrParCtrl	Propiedades del contenido	9.8.2.8.2	S	C→H	Obtiene las condiciones del control parental para el contenido a desaleatorizar.
24	getIccClientInfo	Comunicación entre clientes	9.9.2.4	S	C→H	El **Cliente ECI** lee la identidad y estado de conexión de otro **Cliente ECI** del sistema.
25	getIccMaxClients	Comunicación entre clientes	9.9.2.2	S	C→H	El **Cliente ECI** lee el número máximo de **Clientes ECI** que soporta el **Anfitrión ECI**.
26	getImageTargetId	Seguridad avanzada general	9.5.2.2.17	S	H→C	Obtiene el valor de del Id de Objetivo de Imagen ECI del CPE.
27	getPwrStatus	Gestión de la energía	9.4.7.2.2	S	C→H	Obtiene el valor actual del estado energético.
28	getTime	Reloj	9.4.6.3.2	S	C→H	Lee el reloj del sistema local como un valor entero.
29	reqAsAStartDecryptSession	Desencriptación de seguridad avanzada	9.5.2.3.2	A	C→H	Inicia una sesión de desencriptación en el **intervalo AS** del **Cliente ECI**.
30	reqAsAuthDecrSlotConfig	Desencriptación de seguridad avanzada	9.5.2.3.4	A	H→C	Autentica la configuración del intervalo con mecanismos de autenticación (modo desencriptación).
31	reqAsAuthEncrSlotConfig	Encriptación de seguridad avanzada	9.5.2.5.5	A	C→H	Autentica la configuración de intervalo y los parámetros de encriptación con mecanismos de autenticación (modo encriptación).
32	reqAsClientChalResp	Seguridad avanzada general	9.5.2.2.7	A	C→H	Aplica la clave de autenticación del **Cliente ECI** a los datos y devuelve el resultado.
33	reqAsComputeAkClient	Seguridad avanzada general	9.5.2.2.6	A	C→H	Calcula la clave de autenticación para aplicaciones del **Cliente ECI**.
34	reqAsComputeEncrCw	Encriptación de seguridad avanzada	9.5.2.5.4	A	C→H	Calcula la palabra de control de encriptación.
35	reqAsEventCpChange	Encriptación de seguridad avanzada	9.5.2.5.8	A	H→C	Mensaje de evento cuando se modifican las propiedades de contenido del contenido importado en una sesión de encriptación.
36	reqAsEventSC	Seguridad avanzada general	9.5.2.2.15	A	H→C	Mensaje de evento cuando se modifica el campo control de aleatorización en una sesión.
37	reqAsEventSessionLimit	Seguridad avanzada general	9.5.2.2.12	A	H→C	Se ha alcanzado un valor límite de eventos para el envío de las unidades restantes al **Cliente ECI**.
38	reqAsExportConnEnd	Exportación de seguridad avanzada	9.5.2.4.3	A	C→H	Termina la sesión de exportación en curso.
39	reqAsExportConnSetup	Exportación de seguridad avanzada	9.5.2.4.2	A	C→H	Fija una **Conexión de exportación** de sesión de desencriptación a la de encriptación.
40	reqAsInitSlot	Seguridad avanzada general	9.5.2.2.2	A	C→H	Inicializa el **intervalo AS**.
41	reqAsLdUssk	Seguridad avanzada general	9.5.2.5.6	A	C→H	Carga la clave secreta del **Microservidor**.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
42	reqAsLoadSlotLk	Seguridad avanzada general	9.5.2.2.5	A	C→H	Calcula la clave del enlace de nivel superior (LK1).
43	reqAsMInikLk1	Encriptación de seguridad avanzada	9.5.2.5.7	A	C→H	Calcula el mensaje de inicialización asimétrica de **Microcliente**.
44	reqAsStartEncryptSession	Encriptación de seguridad avanzada	9.5.2.5.3	A	C→H	Inicia una sesión de encriptación.
45	reqAsStopSession	Seguridad avanzada general	9.5.2.2.4	A	C→H	Detiene una sesión.
46	reqCardCmdRes	**Tarjeta inteligente**	9.5.3.6.2	A	C→H	Envía una instrucción a la tarjeta, recibe contestación de la tarjeta.
47	reqCardReInit	**Tarjeta inteligente**	9.5.3.6.3	A	C→H	Reinicia la tarjeta (en caliente o frío) y vuelve a ejecutar la secuencia de inicialización con los últimos valores de las preferencias de inicialización.
48	reqCCardConClose	**Tarjeta inteligente**	9.5.3.5.6	A	H→C	Informa al **Cliente ECI** que se ha cerrado una sesión de tarjeta.
49	reqCCardConOpen	**Tarjeta inteligente**	9.5.3.5.5	A	H→C	Informa al **Cliente ECI** que se ha abierto una sesión de tarjeta.
50	reqCCountry	País	9.4.8.2.2	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** el valor de país preferido real
51	reqCLanguage	Idioma	9.4.8.2.4	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** el valor de idioma preferido real.
52	reqCpChange	Propiedades del contenido	9.8.2.9.2	A	H→C	El **Anfitrión ECI** señaliza que en breve se modificarán las propiedades de contenido del contenido a reencriptar.
53	reqDCAcqModule	Adquisición del carrusel de datos	9.5.4.3	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que adquiera un módulo de carrusel de datos **ECI** específico en un fichero utilizando los parámetros de un filtro de módulo y varios modos.
54	reqDCAcqGroupInfo	Adquisición del carrusel de datos	9.5.4.2	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que lea la estructura GroupInfoIndication en el mensaje DSI del carrusel de datos **ECI** especificado.
55	reqDcrFileQuit	Desencriptación del fichero de medios	9.6.2.4.4.4	A	C→H	El **Cliente ECI** cancela una sesión de desaleatorización con el **Anfitrión ECI**.
56	reqDcrFileData	Petición de datos a través del filtro de ficheros	9.6.2.4.5.2.4	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que adquiera datos a través del Filtro de fichero.
57	reqDcrFileStop	Desencriptación del fichero de medios	9.6.2.4.4.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que detenga la desaleatorización de un **Asa de Medios**.
58	reqDcrFileFilter	Petición de filtro de ficheros	9.6.2.4.5.2.3	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que establezca un filtro de datos para la adquisición de datos de seguridad.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
59	reqDcrFileKeyComp	Petición de cálculo de clave	9.6.2.4.6.3	A	H→C	Inicia cualquier cálculo necesario u otra actividad del **Cliente ECI** disponer de una palabra de control con identificador de clave (Key-ID).
60	reqDcrFileStart	Desencriptación del fichero de medios	9.6.2.4.4.2	A	H→C	Solicita al **Cliente ECI** que desaleatorice o devuelva el estado de desaleatorización de un fichero o un flujo.
61	reqDcrIpServer	Reencriptación	9.7.2.6.5	A	C→H	El **Microcliente** solicita al **Anfitrión ECI** que proporcione la dirección IP del **Microservidor** para comunicaciones adicionales relacionadas con la sesión del **Asa de Medios**.
62	reqDcrMhBcAlloc	Desencriptación del Asa de Medios	9.6.2.2.5	A	C→H	El **Cliente ECI** solicita una sesión del **Asa de Medios** para su acceso a la red de difusión.
63	reqDcrMhCancel	Desencriptación del Asa de Medios	9.6.2.2.6	A	C→H	El **Cliente ECI** cancela una sesión de medios con el **Anfitrión ECI**.
64	reqDcrMhClose	Desencriptación del Asa de Medios	9.6.2.2.4	A	H→C	El **Anfitrión ECI** cierra una sesión de medios con un **Cliente ECI**.
65	reqDcrMhOpen	Desencriptación del Asa de Medios	9.6.2.2.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que abra una sesión de medios de un tipo específico utilizando un **Asa de Medios**.
66	reqDcrMsgRecv	Reencriptación	9.7.2.6.7	A	H→C	El **Anfitrión ECI** proporciona al **Microcliente** un mensaje del **Microservidor** de una sesión de **Asa de Medios**.
67	reqDcrMsgSend	Reencriptación	9.7.2.6.6	A	C→H	El **Microcliente** solicita al **Anfitrión ECI** el envío de un mensaje al **Microservidor** de una sesión de **Asa de Medios**.
68	reqDcrTargetCred	Reencriptación	9.7.2.6.4	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que proporcione los datos de inicialización para una conexión de **Microservidor** normalmente utilizados para la autenticación del objetivo.
69	reqDcrTargets	Reencriptación	9.7.2.6.3	A	H→C	El **Anfitrión ECI** solicita al **Microcliente** que proporcione los objetivos de encriptación para los que puede desencriptar servicios.
70	reqDcrTsData	Reencriptación	9.7.2.6.8	A	C→H	El **Microservidor** proporciona al **Anfitrión ECI** datos que deben ser enviados al **Microcliente** objetivo de un **Asa de Medios** para la desencriptación, incluida información de sincronización conexa del ECM.
71	reqDcrTsDescrquit	Desencriptación de contenido TS	9.6.2.3.4.4	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que termine la desaleatorización de una **sesión del Asa de Medios**.
72	reqDcrTsData	Desencriptación del Microcliente	6.7.2.6.7	A	H→C	El **Anfitrión ECI** proporciona al **Microcliente** datos que serán necesarios próximamente para desencriptar el contenido en el **Asa de Medios**.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
73	reqDcrTsDescrStop	Desencriptación de contenido TS	9.6.2.3.4.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que detenga la desaleatorización de una **sesión de Asa de Medios.**
75	reqDcrTsDescrStart	Desencriptación de contenido TS	9.6.2.3.4.2	A	H→C	Solicita al **Cliente ECI** que desaleatorice o devuelva el estado de desaleatorización de un programa en un TS.
76	reqDcrTsRelocate	Desencriptación del control de fuente TS	9.6.2.3.6.3	A	C→H	Los **Clientes ECI** reubican la fuente del TS.
77	reqDcrTsSection	Desencriptación de contenido TS	9.6.2.3.5.5	A	H→C	Envía una sección adquirida a un **Cliente ECI**
78	reqDcrTsSelectCancel	Desencriptación del control de fuente TS	9.6.2.3.6.6	A	C→H	El **Cliente ECI** cancela su selección previa de programa.
79	reqDcrTsSelectPmt	Desencriptación del control de fuente TS	9.6.2.3.6.5	A	C→H	El **Cliente ECI** selecciona un programa en el TS mediante la PMT.
80	reqDcrTsSelectPrg	Desencriptación del control de fuente TS	9.6.2.3.6.4	A	C→H	El **Cliente ECI** selecciona un programa en el TS mediante el número de programa.
81	reqDcrTsTable	Desencriptación de contenido TS	9.6.2.3.5.6	A	C→H	El **Cliente ECI** adquiere una tabla en el flujo.
82	reqEncrConnDrop	Reencriptación	9.7.2.5.5	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que descarte cualquier información de una conexión de reencriptación anteriormente preautenticada
83	reqEncrConnSetup	Reencriptación	9.7.2.5.4	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que cree una conexión **Objetivo** de reencriptación y que preautentique el **Objetivo** de reencriptación para una ulterior referencia en el establecimiento de una sesión de **Asa de Medios**.
84	reqEncrFileData	Reencriptación	9.7.2.5.18	A	C→H	El **Microservidor** proporciona al **Anfitrión ECI** un mensaje que debe enviarse al **Microcliente Objetivo** de un **Asa de Medios** para desencriptación, incluyendo información de sincronización relacionada con KeyID.
85	reqEncrIpServer	Reencriptación	9.7.2.5.13	A	H→C	El **Anfitrión ECI** solicita la dirección de servidor IP de un **Microservidor** a fin de permitir a los **Microclientes** la creación de conexiones IP.
86	reqEncrMhCancel	Reencriptación	9.7.2.5.9	A	C→H	El **Cliente ECI** finaliza la **Conexión de importación** con el **Cliente ECI** exportador especificado.
87	reqEncrMhClose	Reencriptación	9.7.2.5.8	A	H→C	El **Anfitrión ECI** cierra la **Sesión de reencriptación** con el **Cliente ECI**.
88	reqEncrMhOpen	Reencriptación	9.7.2.5.7	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** la apertura de una sesión de **Asa de Medios** para reencriptar el contenido de una **Conexión de importación** entrante para una conexión de reencriptación establecida.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
89	reqEncrMhQuit	Reencriptación	9.7.2.5.12	A	C→H	El **Cliente ECI** informa al **Anfitrión ECI** que ha terminado la operación de reencriptación del **Asa de Medios**.
90	reqEncrMhStart	Reencriptación	9.7.2.5.10	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que inicie la operación de reencriptación para una sesión de **Asa de Medios**.
91	reqEncrMhStop	Reencriptación	9.7.2.5.11	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que detenga la operación de reencriptación para una sesión de **Asa de Medios**.
92	reqEncrMsgRecv	Reencriptación	9.7.2.5.18	A	H→C	El **Anfitrión ECI** proporciona al **Microservidor** un mensaje procedente de un **Objetivo** de una sesión de **Asa de Medios**.
93	reqEncrMsgSend	Reencriptación	9.7.2.5.14	A	C→H	El **Microservidor** solicita al **Anfitrión ECI** que envíe un mensaje al **Objetivo** de una sesión de **Asa de Medios**.
94	reqEncrTargets	Reencriptación	9.7.2.5.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que proporcione los nodos **Objetivo** que puede autenticar.
95	reqEncrTsData	Reencriptación	9.7.2.5.16	A	C→H	El **Microservidor** proporciona al **Anfitrión ECI** datos que deben enviarse al **Microcliente objetivo** de un **Asa de Medios** para la desencriptación, incluida información de sincronización relacionada con el ECM.
96	reqEncrTsEcm	Reencriptación	9.7.2.5.17	A	C→H	El **Microservidor** genera una sección ECM que el **Microcliente** necesita para labores de desencriptación durante el siguiente criptoperiodo.
97	reqExpConnCancel	Conexión de exportación	9.7.2.3.5	A	C→H	El **Cliente ECI** termina una **Conexión de exportación** con un **Cliente ECI** importador.
98	reqExpConnDrop	Conexión de exportación	9.7.2.3.4	A	H→C	El **Anfitrión ECI** cancela cualquier conexión previamente inicializada de un **Cliente ECI** exportador con un **Cliente ECI** importador.
99	reqExpConnNodes	Conexión de exportación	9.7.2.3.2	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** nodos con la opción de exportación.
100	reqExpConnSetup	Conexión de exportación	9.7.2.3.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que inicialice una **Conexión de exportación** con un **Cliente ECI** importador sobre la base de una **Cadena de importación**.
101	reqExpMhCancel	Conexión de exportación	9.7.2.3.8	A	C→H	El **Cliente ECI** cancela una sesión de exportación.
102	reqExpMhClose	Conexión de exportación	9.7.2.3.7	A	H→C	El **Anfitrión ECI** cierra una sesión de exportación.
103	reqExpMhOpen	Conexión de exportación	9.7.2.3.6	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** que cree una sesión de exportación basada en una **Conexión de exportación** previamente inicializada.
104	reqFileClose	Sistema de ficheros	9.4.5.2.3	A	C→H	Cierra un fichero abierto.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
105	reqFileCreate	Sistema de ficheros	9.4.5.4.3	A	C→H	Crea un nuevo fichero.
106	reqFileDelete	Sistema de ficheros	9.4.5.4.4	A	C→H	Suprime un fichero.
107	reqFileDir	Sistema de ficheros	9.4.5.4.5	A	C→H	Enumera los nombres de los ficheros disponibles en el sistema de ficheros de **Clientes ECI**.
108	reqFileOpen	Sistema de ficheros	9.4.5.2.2	A	C→H	Abre un fichero privado de un **Cliente ECI**.
109	reqFileRead	Sistema de ficheros	9.4.5.3.3	A	C→H	Lee bytes consecutivos comenzando por la ubicación del fichero actual.
110	reqFileRemoveData	Sistema de ficheros	9.4.5.3.5	A	C→H	Suprime datos de un fichero en la ubicación actual.
111	reqFileSeek	Sistema de ficheros	9.4.5.3.4	A	C→H	Reposiciona la ubicación actual del fichero.
112	reqFileStat	Sistema de ficheros	9.4.5.4.2	A	C→H	Devuelve el tamaño y la hora de modificación del fichero.
113	reqFileWrite	Sistema de ficheros	9.4.5.3.2	A	C→H	Escribe bytes consecutivos empezando en la ubicación actual del fichero.
114	reqHCardConClose	**Tarjeta inteligente**	9.5.3.5.7	A	C→H	Informa al **Anfitrión ECI** que el **Cliente ECI** desea terminar una sesión con la tarjeta conectada.
115	reqHCountry	País	9.4.8.2.1	A	C→H	Solicita los valores preferidos de país del **Anfitrión ECI**.
116	reqHLanguage	Idioma	9.4.8.2.3	A	C→H	Solicita los valores preferidos de idioma del **Anfitrión ECI**.
117	reqHttpGetData	HTTP Get	9.4.4.6.3	A	C→H	Realiza una petición HTTP Get a un URL y transfiere el resultado como datos al cliente.
118	reqHttpGetFile	HTTP Get	9.4.4.6.3	A	C→H	Realiza una petición HTTP Get a un URL y almacena el resultado en un fichero.
119	reqIccPipeCancel	Comunicación entre clientes	9.9.2.7	A	C→H	El **Cliente ECI** cancela el conducto.
120	reqIccPipeClose	Comunicación entre clientes	9.9.2.8	A	H→C	El **Anfitrión ECI** informa al **Cliente ECI** que se ha cerrado el conducto con el par.
121	reqIccPipeMsgRecv	Comunicación entre clientes	9.9.2.10	A	H→C	El **Cliente ECI** recibe un mensaje de su par de un conducto.
122	reqIccPipeMsgSend	Comunicación entre clientes	9.9.2.9	A	C→H	El **Cliente ECI** envía un mensaje a su par de un conducto.
123	reqIccPipeOpen	Comunicación entre clientes	9.9.2.5	A	C→H	Solicita la apertura de un conducto con otro **Cliente ECI**.
124	reqIccPipeOpenReq	Comunicación entre clientes	9.9.2.6	A	H→C	Petición entrante de otro **Cliente ECI** para la apertura de un conducto.
125	reqIccSystemReady	Comunicación entre clientes	9.9.2.3	A	H→C	El **Anfitrión ECI** informa al **Cliente ECI** que se han inicializado todos los **Clientes ECI**.
126	reqImpConnCancel	Conexión de importación	9.7.2.4.6	A	C→H	El **Cliente ECI** termina la **Conexión de importación** con el **Cliente ECI** exportador especificado.
127	reqImpConnChain	Conexión de importación	9.7.2.4.3	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** importador que proporcione una cadena de entrada para un nodo de importación específico.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
128	reqImpConnChainRenew	Conexión de importación	9.7.2.4.3	A	C→H	El **Cliente ECI** solicita al **Anfitrión ECI** que reinicialice la conexión utilizando una **Cadena de importación** actualizada.
129	reqImpConnDrop	Conexión de importación	9.7.2.4.5	A	H→C	El **Anfitrión ECI** descarta la **Conexión de importación** con el **Cliente ECI** exportador especificado.
130	reqImpConnNodes	Conexión de importación	9.7.2.4.2	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** importador que proporcione sus nodos de importación.
131	reqImpConnSetup	Conexión de importación	9.7.2.4.4	A	H→C	El **Anfitrión ECI** solicita al **Cliente ECI** importador que inicialice una **Conexión de importación** con un **Cliente ECI** exportador específico a través de un nodo de importación.
132	reqIpAddrinfo	Conectores IP	9.4.4.3.4	A	C→H	Obtiene la dirección de un **Anfitrión ECI** (distante).
133	reqIpClose	Conectores IP	9.4.4.3.3	A	C→H	Cierra el conector IP **ECI**.
134	reqIpSocket	Conectores IP	9.4.4.3.2	A	C→H	Abre un conector IP **ECI**.
135	reqIpTcpAccept	Conector TCP/IP	9.4.4.5.5	A	C→H	El servidor TCP par acepta la conexión del cliente TCP par.
136	reqIpTcpConnect	Conector TCP/IP	9.4.4.5.2	A	C→H	El cliente TCP se conecta al servidor TCP par.
137	reqIpTcpRecv	Conector TCP/IP	9.4.4.5.4	A	C→H	Recibe datos del par conectado.
138	reqIpTcpSend	Conector TCP/IP	9.4.4.5.3	A	C→H	Envía datos al par conectado.
139	reqIpUdpRecvMsg	Conector UDP/IP	9.4.4.4.3	A	C→H	Recibe un mensaje del puerto UDP par.
140	reqIpUdpSendMsg	Conector UDP/IP	9.4.4.4.2	A	C→H	Envía un mensaje al puerto UDP par.
141	reqParAuthChk	Propiedades del contenido	9.8.2.10.3	A	C→H	Solicita al **Anfitrión ECI** que realice una autenticación parental en nombre del **Cliente ECI**.
142	reqParAuthChkCan	Propiedades del contenido	9.8.2.10.4	A	C→H	Cancela una petición previa de autenticación parental al Anfitrión.
143	reqParAuthCid	Propiedades del contenido	9.8.2.10.5	A	H→C	Solicita la autorización del código pin parental para un (futuro) elemento de contenido a decodificar. Ello puede activar un diálogo de autenticación parental.
144	reqParAuthDel	Propiedades del contenido	9.8.2.11.2	A	H→C	El **Anfitrión ECI** delega una autenticación parental a un **Cliente ECI**.
145	reqParAuthDelCan	Propiedades del contenido	9.8.2.11.3	A	H→C	El **Anfitrión ECI** cancela una petición de autenticación parental previa al **Cliente ECI**.
146	reqPwrChange	Gestión de energía	9.4.7.2.4	A	H→C	Notificación de cambio del estado energético.
147	reqTimerCancel	Temporizador	9.4.6.2.3	A	C→H	Cancela un evento previo de fijación del temporizador.
148	reqTimerEvent	Temporizador	9.4.6.2.2	A	C→H	Fija un evento futuro del temporizador.
149	reqUiClientQuery	Interfaz de Usuario	9.4.3.4.8	A	H→C	El **Cliente ECI** recibe una petición de la aplicación HTML del navegador y proporciona una contestación (dinámica).

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
150	reqUiContainerMount	Interfaz de Usuario	9.4.3.4.2	A	C→H	Crea un contenedor de aplicaciones de la Interfaz de Usuario (UI) con recursos HTML para soportar sesiones UI.
151	reqUiSessionCancel	Interfaz de Usuario	9.4.3.4.7	A	H→C	El **Anfitrión ECI** cancela una sesión de la interfaz de **Usuario**.
152	reqUiSessionClose	Interfaz de Usuario	9.4.3.4.6	A	C→H	El **Cliente ECI** finaliza una sesión de la interfaz de **Usuario**.
153	reqUiSessionCommence	Interfaz de Usuario	9.4.3.4.4	A	H→C	El **Anfitrión ECI** propone al **Cliente ECI** la apertura de una sesión de interfaz de usuario (UI).
154	reqUiSessionOpen	Interfaz de Usuario	9.4.3.4.5	A	C→H	El **Cliente ECI** solicita la apertura de una interfaz de **Usuario** con el **Usuario** y el contenido en pantalla.
155	reqPwrWakeupEvent	Gestión de la energía	9.4.7.3	A	H→C	Señaliza la expiración del temporizador despertador.
156	setApiVersion	Descubrimiento de la interfaz	9.4.2.4	S	C→H	Fija la versión de la API de anfitrión a utilizar.
157	setAsPermitCPChange	Seguridad avanzada, encriptación	9.5.2.4	S	C→H	Habilita/inhabilita cambios en las propiedades del contenido (CP) importados que afectan a la selección de la palabra de control para encriptación en una sesión de encriptación.
158	setAsSC	Seguridad avanzada, encriptación	9.5.2.4	S	C→H	Fija el campo control de aleatorización del contenido encriptado de una sesión de encriptación.
159	setAsSessionLimitEvent	Seguridad avanzada, generalidades	9.5.2.5.11	S	C→H	Fija el valor límite para el envío de un mensaje reqAsEventSessionLimit al **Cliente ECI**
160	setCardMatch	**Tarjeta inteligente**	9.5.3.5.2	S	C→H	Fija la lista de especificadores de identificación de tarjetas para el **Cliente ECI**.
161	setCpSync	Propiedades del contenido	9.8.2	S	C→H	El **Cliente ECI** señaliza que el conjunto actual de propiedades del contenido es consistente y puede aplicarse al contenido a desaleatorizar mediante la siguiente palabra de control.
162	setDcrBasicUri	Propiedades del contenido	9.8.2.5.1	S	C→H	Fija la URI básica para el contenido a desaleatorizar.
163	setDcrCustUri	Propiedades del contenido	9.8.2.4.1	S	C→H	Fija la URI a medida para el contenido a desaleatorizar.
164	setDcrMarkBasic	Propiedades del contenido	9.8.2.7.5	S	C→H	Fija la carga útil de marcaje básica para el contenido a desaleatorizar.
165	setDcrMarkExt	Propiedades del contenido	9.8.2.7.6	S	C→H	Fija la carga útil de marcaje ampliada para el contenido a desaleatorizar.
166	setDcrMarkMeta	Marca de agua	9.8.2.7.3	S	C→H	Fija un valor de control del sistema de marcaje.
167	setDcrMhMatch	Desencriptación de Asa de medios	9.6.2.2.2	S	C→H	Señaliza al **Anfitrión ECI** los identificadores que permiten reconocer al **Cliente ECI** para la desaleatorización del contenido.

Cuadro I-2 – Lista de todos los mensajes de las API en orden alfabético

N.°	Mensaje	API	Cláusula	Tipo	Dir.	Descripción
168	setDcrModes	Reencriptación	9.7.2.6.1	S	C→H	El **Microcliente** informa al **Anfitrión ECI** sobre los modos que soporta (modos de encriptación, modos de formato de datos y modos de sincronización).
170	setDcrOutputCtl	Propiedades del contenido	9.8.2.6.1	S	C→H	Fija restricciones del control de salida para el contenido a desaleatorizar.
171	setDcrParCtl	Propiedades del contenido	9.8.2.8.1	S	C→H	Fija condiciones de control parental para el contenido a desaleatorizar.
172	setDcrStdUri	Propiedades del contenido	9.8.2.8.1	S	C→H	Fija la URI estándar para el contenido a desaleatorizar.
173	setDcrTsSectionAcq	Adquisición de datos TS para desencriptación	9.6.2.3.5.4	S	C→H	Fija un filtro para la adquisición de secciones.
176	setDcrTsSectionAcqDefault	Adquisición de datos TS para desencriptación	9.6.2.3.5.3	S	C→H	Fija un filtro por defecto para la adquisición de secciones.
177	setEncrModes	Reencriptación	9.7.2.5.2	S	C→H	El **Microservidor** informa al **Anfitrión ECI** sobre los modos que soporta (modos de encriptación, modos de formato de datos y modos de sincronización).
178	setPwrInfo	Gestión de energía	9.4.7.2.3	S	C→H	Solicita notificaciones de eventos relativos a cambios del estado energético.
179	setUiClientAttention	Interfaz de Usuario	9.4.3.4.3	S	C→H	El **Cliente ECI** indica su deseo de iniciar una sesión UI no asociada a un **Asa de Medios**.
180	setPwrWakeup	Gestión de energía	9.4.7.3	S	C→H	Fija la hora del despertador para el **Cliente ECI**.

Apéndice II

Aspectos que se han de mejorar

(Este apéndice no forma parte integrante de la presente Recomendación.)

Se ha llegado a la conclusión de que es necesario mejorar y validar la presente Recomendación para que cumpla los requisitos estipulados en [UIT-T J.1010] y que [UIT-T J.1010] debe actualizarse para que incorporar los requisitos de la especificación de protección mejorada del contenido (ECP) de MovieLabs [b-ECP]. Las Recomendaciones [UIT-T J.1011], [UIT-T J.1012], [UIT-T J.1013], [UIT-T J.1014], [UIT-T J.1015] y [b-UIT-T J.1015.1] deben actualizarse en el futuro para incorporar esas modificaciones de la [UIT-T J.1010].

Varios Estados Miembros de la UIT y otras partes interesadas de diversas industrias – incluidos fabricantes de dispositivos y componentes electrónicos, los propietarios y titulares de contenido protegido por derecho de autor, proveedores de servicios de televisión por satélite (OTT) y de televisión por cable, y los proveedores de soluciones de sistemas de acceso condicional (CAS) y de gestión de derechos digitales (DRM) – de todo el mundo han expresado su preocupación por el hecho de que la interfaz común integrada (ECI) no satisface plenamente los requisitos de la ECP ni los requisitos más amplios de protección del contenido de la industria.

Más concretamente, sus preocupaciones se plantearon en las contribuciones a la reunión de la Comisión de Estudio 9 (CE 9) del UIT-T (16 a 23 de abril de 2020). En las contribuciones de Israel, Australia, el Miembro de Sector del UIT-T Samsung, y los Asociados de la CE 9, Sky Group y MovieLabs, se propuso la introducción de diversas modificaciones en las Recomendaciones sobre ECI, pero no se llegó a un acuerdo al respecto. Estas modificaciones se consignaron en [b-SG9 Report 17 Ann.1].

Las propuestas tienen por objeto:

1) Simplificar el sistema ECI reduciendo su alcance.
2) Eliminar el DRM.
3) Eliminar la recodificación del contenido.
4) Eliminar la gestión de software.
5) Añadir API para el almacenamiento seguro y las operaciones criptográficas.
6) Permitir escalas de claves específicas para cada proveedor.
7) Utilizar los requisitos de la norma UIT-T J.1207 TEE.
8) Incluir la implementación de la TEE para la VM.
9) Mejorar la robustez de los algoritmos criptográficos, por ejemplo, utilizando SHA-384.
10) Utilizar certificados normalizados, como los de la Recomendación UIT-T X.509.
11) Reconsiderar las comunicaciones entre clientes.
12) Realizar declaraciones de coordinación adicionales con el ETSI.
13) Realizar revisiones adicionales por pares.
14) Analizar alternativas al modelo de autoridad fiduciaria.
15) Definir más detalladamente los aspectos técnicos de las normas de cumplimiento y robustez de ECI.
16) Añadir requisitos de diversidad, por ejemplo, la aleatorización del espacio de direcciones.
17) Añadir requisitos para verificar la integridad en tiempo de ejecución.

Estas propuestas responden a que la protección de contenidos y las amenazas de su compromiso evolucionan continuamente. La ECI fue concebida originalmente casi una década antes de que se aprobara de esta Recomendación UIT-T. Los sistemas como la ECI han de evaluarse regularmente respecto del estado actual de la técnica tanto en lo relativo a las técnicas analíticas como a los requisitos de protección de la industria.

Existen otros mecanismos que permiten la interoperabilidad. En particular, en el caso de la utilización de DRM, la mayoría de los servicios de vídeo por Internet han desplegado otras soluciones que ofrecen interoperabilidad y satisfacen sus necesidades.

Es importante que haya una mayor claridad, por cuanto muchos Estados Miembros consideran que las normas de la UIT tienen gran influencia en el desarrollo de sus mercados e industrias. La lista de preocupaciones garantiza la implementación de ECI en sus mercados nacionales, lo que puede requerir comprender plenamente las repercusiones de la presente Recomendación UIT-T y velar por que se tomen en consideración dichas cuestiones cuando se examine la legislación o la reglamentación o cuando las necesidades del mercado exijan que los aparatos de televisión digital de consumo sean interoperables. También garantiza que los fabricantes de equipo tecnológico, que pueden preferir utilizar un conjunto único de requisitos u otras normas a la hora de diseñar los productos, puedan tener en cuenta estas cuestiones al desarrollar productos para diferentes mercados.

Bibliografía

[b-UIT-T J.1015.1] Recomendación UIT-T J.1015.1 (2020), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; Sistema de seguridad avanzada – Bloque de escalera de claves: Autentificación de la información de reglas de utilización de palabra de control y datos conexos 1*.

[b-UIT-T J Supl. 7] Suplemento 7 a las Recomendaciones de la serie J del UIT-T (2020), *Interfaz común integrada (ECI) para soluciones CA/DRM intercambiables; Directrices para la implementación de ECI*.

[b-SG9 Report 17 Ann.1] Informe de la reunión de la CE 9 del UIT-T, SG9-R17-Anexo 1 (2020), Anexo 1 al Informe 17 de la reunión virtual de la CE 9 celebrada del 16 al 23 de abril de 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, *RESTfull Web services*, L. o'Reilly, 2007.

[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 800-52r1] NIST SP 800-52 rev2 (Agosto de 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).
Disponible en 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, *High Bandwidth Digital Content Protection System*, *Mapping HDCP to HDMI*, Revisión 2.3, 28 de febrero de 2018
https://www.digital-cp.com/sites/default/files/HDCP%20on%20HDMI%20Specification%20Rev2_3.pdf

[b-Illgner] Klaus Illgner, Christoph Schaaf, Marnix Vlot, *Embedded Common Interface (ECI) for Digital Broadcasting Applications: Security and Interoperability combined*, Broadband Journal of the SCTE, Vol. 38, N.° 3, agosto de 2016.

[b-Menezes] Menezes, A., van Oorschot, P. and Vanstone, S, *Handbook of Applied Cryptography*, CRC Press, 1996.

[b-ECP] MovieLabs Specification for Enhanced Content Protection – Versión 1.2
https://movielabs.com/ngvideo/MovieLabs_ECP_Spec_v1.2.pdf

Aunque los hiperenlaces incluidos en esta cláusula sean válidos en el momento de su publicación, no es posible garantizar su validez a largo plazo.

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie D	Principios de tarificación y contabilidad y cuestiones económicas y políticas de las telecomunicaciones/TIC internacionales
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedia
Serie I	Red digital de servicios integrados
Serie J	**Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedia**
Serie K	Protección contra las interferencias
Serie L	Medio ambiente y TIC, cambio climático, ciberdesechos, eficiencia energética, construcción, instalación y protección de los cables y demás elementos de planta exterior
Serie M	Gestión de las telecomunicaciones, incluida la RGT y el mantenimiento de redes
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de la transmisión telefónica, instalaciones telefónicas y redes de líneas locales
Serie Q	Conmutación y señalización, y mediciones y pruebas asociadas
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos, comunicaciones de sistemas abiertos y seguridad
Serie Y	Infraestructura mundial de la información, aspectos del protocolo Internet, redes de próxima generación, Internet de las cosas y ciudades inteligentes
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Impreso en Suiza
Ginebra, 2020