Международный союз электросвязи

МСЭ-Т J.1012

(04/2020)

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

СЕРИЯ J: КАБЕЛЬНЫЕ СЕТИ И ПЕРЕДАЧА СИГНАЛОВ ТЕЛЕВИЗИОННЫХ ЗВУКОВЫХ ПРОГРАММ И ДРУГИХ МУЛЬТИМЕДИЙНЫХ СИГНАЛОВ

Условный доступ и защита – Заменяемые встроенные решения для обеспечения условного доступа и управления цифровыми правами

Встроенный общий интерфейс для заменяемых решений CA/DRM; контейнер, загрузчик, интерфейсы, аннулирование CA/DRM

Рекомендация МСЭ-Т J.1012

ITU-T

Международный союз электросвязи

Рекомендация МСЭ-Т J.1012

Встроенный общий интерфейс (ECI) для заменяемых решений CA/DRM; контейнер, загрузчик, интерфейсы, аннулирование CA/DRM

Резюме

Рекомендация МСЭ-Т J.1012 входит в состав состоящего из нескольких частей итогового документа и содержит спецификацию контейнера, загрузчика, интерфейсов, аннулирования, которая касается встроенного общего интерфейса для заменяемых решений CA/DRM.

Настоящая Рекомендация МСЭ-Т является переложением стандарта ETSI GS ECI 001-3 и представляет собой результат сотрудничества ИК9 МСЭ-Т и ETSI ISG ECI. Изменения внесены в разделы 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 и I-2, а также в раздел "Библиография". Понадобились также некоторые редакторские правки.

Хронологическая справка

Издание	Рекомендация	Утверждение	Исследовательская комиссия	Уникальный идентификатор*
1.0	МСЭ-Т J.1012	23.04.2020 года	9-я	11.1002/1000/13573

Ключевые слова

CA, DRM, замена.

* Для получения доступа к Рекомендации наберите в адресном поле вашего браузера URL http://handle.itu.int/, после которого следует уникальный идентификатор Рекомендации. Например, http://handle.itu.int/11.1002/1000/11830-en.

ПРЕДИСЛОВИЕ

Международный союз электросвязи (МСЭ) является специализированным учреждением Организации Объединенных Наций в области электросвязи и информационно-коммуникационных технологий (ИКТ). Сектор стандартизации электросвязи МСЭ (МСЭ-Т) – постоянный орган МСЭ. МСЭ-Т отвечает за изучение технических, эксплуатационных и тарифных вопросов и за выпуск Рекомендаций по ним с целью стандартизации электросвязи на всемирной основе.

На Всемирной ассамблее по стандартизации электросвязи (ВАСЭ), которая проводится каждые четыре года, определяются темы для изучения исследовательскими комиссиями МСЭ-Т, которые, в свою очередь, вырабатывают Рекомендации по этим темам.

Утверждение Рекомендаций МСЭ-Т осуществляется в соответствии с процедурой, изложенной в Резолюции 1 ВАСЭ.

В некоторых областях информационных технологий, которые входят в компетенцию МСЭ-Т, необходимые стандарты разрабатываются на основе сотрудничества с ИСО и МЭК.

ПРИМЕЧАНИЕ

В настоящей Рекомендации термин "администрация" используется для краткости и обозначает как администрацию электросвязи, так и признанную эксплуатационную организацию.

Соблюдение положений данной Рекомендации осуществляется на добровольной основе. Однако данная Рекомендация может содержать некоторые обязательные положения (например, для обеспечения функциональной совместимости или возможности применения), и в таком случае соблюдение Рекомендации достигается при выполнении всех указанных положений. Для выражения требований используются слова "следует", "должен" (shall) или некоторые другие обязывающие выражения, такие как "обязан" (must), а также их отрицательные формы. Употребление таких слов не означает, что от какой-либо стороны требуется соблюдение положений данной Рекомендации.

ПРАВА ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

МСЭ обращает внимание на вероятность того, что практическое применение или выполнение настоящей Рекомендации может включать использование заявленного права интеллектуальной собственности. МСЭ не занимает какую бы то ни было позицию относительно подтверждения, действительности или применимости заявленных прав интеллектуальной собственности независимо от того, доказываются ли такие права членами МСЭ или другими сторонами, не относящимися к процессу разработки Рекомендации.

На момент утверждения настоящей Рекомендации МСЭ получил извещения об интеллектуальной собственности, защищенной патентами, которые могут потребоваться для выполнения настоящей Рекомендации. Однако те, кто будет применять Рекомендацию, должны иметь в виду, что вышесказанное может не отражать самую последнюю информацию, и поэтому им настоятельно рекомендуется обращаться к патентной базе данных БСЭ по адресу: http://www.itu.int/ITU-T/ipr/.

СОДЕРЖАНИЕ

Введение

Настоящая Рекомендация МСЭ-Т[1] является переложением стандарта ЕТСИ [b-ETSI GS ECI 001-3] и представляет собой результат сотрудничества ИК9 МСЭ-Т и ETSI ISG ECI. Изменения внесены в разделы 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 и I-2, а также в раздел "Библиография". Понадобились также некоторые редакторские правки.

Настоящая Рекомендация призвана способствовать повышению функциональной совместимости и развитию конкуренции в сфере услуг электросвязи – в частности, на рынке радиовещательных и аудиовизуальных устройств. В зависимости от ситуации в Государствах-Членах это может быть целесообразно и полезно и в отношении других существующих технологий.

Защита услуг и контента, реализуемая в системах условного доступа (CA) и управления цифровыми правами (DRM), крайне важна в условиях динамичного развития сферы услуг цифрового радиовещания и широкополосной связи. Эта задача включает в себя распределение контента высокой четкости (HD) и сверхвысокой четкости (UHD) по различным типам оборудования в помещении клиента (**CPE**)[2] для защиты бизнес-моделей владельцев контента и поставщиков услуг, включая радиовещательные организации и операторов платного телевидения (PayTV). Системы CA применяются главным образом для защиты контента, распространяемого по однонаправленным сетям, которые обычно используются в вещательной среде. В отличие от них системы DRM базируются на двунаправленных сетях и разрешают авторизованным пользователям доступ к контенту на сертифицированных устройствах, как правило, с предоставлением прав в плане богатого выбора контента. В настоящей Рекомендации используется термин "системы CA/DRM", так как на практике не всегда можно четко определить различие между функциональными возможностями CA и DRM.

Реализованные в настоящее время решения CA/DRM, будь то встроенные или съемные устройства, зачастую приводят к пользовательским ограничениям как для поставщиков услуг и платформ, так и для потребителей. В результате возникает зависимость потребителей от действующей сети, поставщиков услуг и контента, а также используемого оборудования **CPE**, поддерживающего классическое цифровое радиовещание, услуги ТВ с использованием протокола Интернет (IPTV) или на основе технологии over-the-top (OTT). В то время как устройства **CPE** со встроенными проприетарными функциональными возможностями CA и DRM привязывают потребителя к оператору конкретной платформы, съемные аппаратные модули позволяют использовать устройства **CPE** серийного производства, например абонентские приставки (STB) и интегрированные ТВ-приемники (iDTV). Конструктивные характеристики и стоимость съемных аппаратных модулей не соответствуют будущим запросам рынка, особенно касающимся просмотра защищенного контента на планшетах и мобильных устройствах, а также в тех случаях, когда стоимость оборудования является критичной.

Таким образом, существующие технологии ограничивают свободу действий для многих участников рынка цифрового мультимедийного контента. В связи с техническим прогрессом становятся осуществимыми инновационные решения CA/DRM на основе программного обеспечения. Эти решения, позволяя достичь максимальной совместимости при сохранении высокого уровня безопасности, дают надежды на удовлетворение растущих потребностей рынка, стимулирование развития новых направлений бизнеса, а также предоставление широкого доступа потребителей к контенту через широковещательные и широкополосные соединения.

Потребители заинтересованы в том, чтобы устройства **CPE**, приобретенные ими для собственных нужд, могли в дальнейшем использоваться после ухода или смены поставщика сети и применяться для получения услуг различных коммерческих видеопорталов. Это может быть достигнуто путем внедрения функционально совместимых механизмов CA и DRM в устройствах **CPE** на основе надлежащей архитектуры безопасности. Избежать дальнейшей фрагментации на рынке оборудования **CPE** и содействовать конкуренции можно лишь посредством решений, учитывающих интересы потребителей и обеспечивающих гибкую процедуру взаимозаменяемости систем CA и DRM, связанных с современной безопасной средой.

---

[1] В Дополнении II определен ряд тематических областей для доработки.

[2] Полужирным шрифтом в тексте настоящей Рекомендации выделены термины, определения которых в контексте встроенного общего интерфейса могут отличаться от общеупотребительных.

Оператор платформы заинтересован в гибком развертывании и эффективном управлении технологиями обеспечения безопасности в различных сетях и на всех типах устройств. Бесшовный способ обновления существующих устройств с новейшими системами безопасности предоставляет беспрецедентные возможности для бизнеса.

Экосистема ECI, как указано в настоящей Рекомендации и согласно документации по **ECI**, состоящей из нескольких частей, обеспечивает такие важные характеристики, как гибкость и масштабируемость благодаря реализации на базе программного обеспечения, а также взаимозаменяемость, позволяющая применять перспективные решения и внедрять инновации. Дополнительным аспектом является применимость для распространения контента через различные типы сетей, включая классическое цифровое радиовещание, услуги технологий IPTV и OTT. Спецификация системы **ECI** как открытой экосистемы, стимулирующей развитие рынка, закладывает основу для взаимозаменяемости систем CA и DRM в оборудовании **CPE**, обеспечивая минимально возможные затраты для потребителей и минимальные ограничения для поставщиков систем CA или DRM при продвижении намеченной к выпуску продукции на рынке платного телевидения.

Настоящая Рекомендация представляет собой часть 3 этого состоящего из нескольких частей документа, в которой дано определение всех необходимых элементов, важных для целей загрузки и замены клиентов CA/DRM (**клиентов ECI**) и их среды исполнения (**хост ECI**) в надежной среде, включая связь с необходимыми функциональными объектами через подробно описываемые здесь интерфейсы API. Часть 4 того же документа посвящена виртуальной машине, а часть 5 – усовершенствованной системе безопасности.

Рекомендация МСЭ-Т J.1012

Встроенный общий интерфейс для заменяемых решений CA/DRM; контейнер, загрузчик, интерфейсы, аннулирование CA/DRM

1 Сфера применения

Архитектура **системы ECI** определена в [ITU-T J.1011]; см. также [b-ETSI GS ECI 001-1]. Система **ECI** работает на основе требований, определенных в [ITU-T J.1010]; см. также [b-ETSI GS ECI 001-2]. В настоящей Рекомендации определены базовые функциональные возможности **экосистемы ECI**, в том числе дается подробная информация о контейнере, загрузчике, интерфейсах и аннулировании CA/DRM; см. также [b-Illgner]. Основным преимуществом и инновацией **экосистемы ECI** по сравнению с действующими системами является полноценная, основанная на программном обеспечении архитектура загрузки и замены систем CA/DRM без применения съемных аппаратных модулей. Контейнеры программного обеспечения гарантируют наличие безопасной среды ("песочницы") для ядер CA и DRM, далее именуемых **клиентами ECI**, и их собственных копий **виртуальной машины**. Необходимые и подходящие интерфейсы прикладного программирования (API) между **клиентами ECI** и **хостом ECI** гарантируют, что несколько **клиентов ECI** могут действовать в безопасной рабочей среде и быть полностью изолированными от остального встроенного программного обеспечения **CPE**. Приведено подробное описание интерфейсов API. Установка и замена **хоста ECI**, а также нескольких **клиентов ECI** – это задача загрузчика **ECI**, который на начальном этапе загружается загрузчиком микросхем. **Хост ECI** и **клиенты ECI** загружаются с помощью карусели данных цифрового телевизионного радиовещания (DVB) для вещательных услуг и/или механизмов на основе IP с сервера при использовании широкополосного доступа. Этот процесс протекает в защищенной и безопасной среде и обеспечивает иерархию доверия для установки и замены **хоста ECI** и **клиентов ECI**. Таким образом осуществляется эффективная защита от атак, направленных на нарушение целостности, и от действий посредством замены. В связи с этим в **экосистему ECI** внедрен усовершенствованный механизм безопасности, основанный на эффективной и углубленной обработке управляющих слов (CW). Данный механизм называется **блоком лестницы ключей (**многоступенчатая система шифрования) и интегрируется в оборудование с однокристальными системами (SoC) в целях обеспечения максимальной безопасности, необходимой для соответствия спецификациям **ECI**. Кроме того, функции усовершенствованной системы безопасности, относящиеся конкретно к **ECI**, играют ключевую роль в процессе повторного шифрования при сохранении защищенного контента и/или при экспорте защищенного контента на внешние устройства, совместимые или несовместимые с интерфейсом **ECI**. Усовершенствованная микросистема DRM обеспечивает необходимый набор функций и является неотъемлемой частью данной концепции. Функциональные возможности усовершенствованной системы безопасности применимы также в случае аннулирования оборудования **CPE** или определенного **клиента ECI**. Соответствующие интерфейсы API определены в настоящей Рекомендации. Усовершенствованная система безопасности подробно описывается в [ITU-T J.1014] и [ITU-T J.1015]; см. также [b-ETSI GS ECI 001-5-1] и [b-ETSI GS ECI 001-5-2].

Ряд интерфейсов API характеризуют **экосистему ECI** и гарантируют соединение с соответствующими взаимосвязанными объектами, например с загрузчиками **ECI**, импортом и экспортом защищенного контента, усовершенствованной системой безопасности, дешифрованием и шифрованием, локальными хранилищами и защитой водяными знаками. Дополнительные интерфейсы API доступны для **клиентов ECI** с поддержкой человеко-машинного интерфейса (MMI) или для дополнительных устройств чтения **смарт-карт**.

Замена **клиентов ECI** инициируется **пользователем** или может быть запрошена **оператором** при необходимости установки обновлений. В течение периода доступности локального хранилища на персональном видеомагнитофоне (PVR), а также при необходимости экспорта данных поддерживаются как минимум два **клиента ECI** и два дополнительных **клиента ECI**.

В следующих разделах настоящей Рекомендации представлены подробные спецификации.

В разделе 5 описывается система сертификации **ECI**, охватывающая **сертификаты**, предназначенные для **загрузчика хоста ECI**, **загрузчика клиента ECI** и **сертификатов операторов ECI**, в том числе

определение этих **сертификатов** и соответствующий **список аннулирования**, их построение в цепочки и структура **корневого сертификата**.

В разделе 6 рассматривается **загрузчик хоста ECI**. Процесс загрузки **хоста ECI** управляет хранением образа, проверкой подлинности образа при помощи оборудования **CPE** с использованием аутентификационных данных, предоставленных **доверительным органом ECI**, и последующей активацией образа. Сюда входят спецификация формата файлов, транспортный протокол и аннулирование **образов хоста ECI** для конкретного **оператора**.

В разделе 7 приведена подробная спецификация, касающаяся **загрузчика клиента ECI** и основанная на том, что **хост ECI** способен загружать, хранить и активировать **образы клиента ECI** и сопутствующие данные. Процесс загрузки **клиента ECI** можно разделить на несколько этапов, первый из которых – это процесс обнаружения, а последний – загрузка и инициализация **клиентов ECI**, что позволяет выполнять процесс загрузки, используя данные из вещательного потока или из интернета.

В разделе 8 подробно описывается спецификация аннулирования, в том числе функциональная возможность выборочного исключения услуг для оборудования **CPE** на основе предоставленного **доверительным органом ECI** статуса аппаратной части **CPE**, хоста **ECI**, других **систем управления платформой** и загружаемых **клиентов ECI**.

В разделе 9 приведены подробные спецификации интерфейсов **клиентов ECI**, включающие в себя исчерпывающее описание элементов, необходимых для экосистемы **ECI**: интерфейсы API для общих ресурсов **хоста ECI**, связанные с **ECI** ресурсы **хоста ECI**, ресурсы дешифрования **хоста ECI**, ресурсы повторного шифрования **хоста ECI**, ресурсы, связанные с защитой контента, и ресурсы для соединения **"клиент ECI – клиент ECI"**.

В разделе 10 определены обязательные и дополнительные функциональные возможности **хоста ECI**.

Указанная базовая спецификация **ECI** применяется только для приема и последующей обработки контента, который контролируется системами условного доступа и/или управления цифровыми правами и зашифрован поставщиком услуг

Контент, не контролируемый системой условного доступа и/или системой DRM, в настоящей Рекомендации не рассматривается.

Настоящая Рекомендация предназначена для использования совместно с договорной базой (лицензионным соглашением), правилами соответствия и обеспечения устойчивости, а также соглашением об установленном процессе сертификации под контролем доверительного органа, которые не подпадают под технические спецификации, включенные в групповые спецификации **ECI**. Некоторые из указанных основных положений приведены в информационном приложении к спецификации [b-ETSI GS ECI 001-6], касающейся безопасной среды, которое содержит описания технических механизмов и взаимосвязи элементов в безопасной среде.

2 Справочные документы

Указанные ниже Рекомендации МСЭ-Т и другие справочные документы содержат положения, которые путем ссылок на них в данном тексте составляют положения настоящей Рекомендации. На момент публикации указанные издания были действующими. Все Рекомендации и другие справочные документы могут подвергаться пересмотру; поэтому всем пользователям данной Рекомендации предлагается изучить возможность применения последнего издания Рекомендаций и других справочных документов, перечисленных ниже. Перечень действующих на настоящий момент Рекомендаций МСЭ-Т регулярно публикуется. Ссылка на документ, приведенный в данной Рекомендации, не придает ему как отдельному документу статус Рекомендации.

[ITU-T J.1010] Рекомендация МСЭ-Т J.1010 (2016 год), *Встроенный общий интерфейс для заменяемых решений CA/DRM; сценарии использования и требования*.

[ITU-T J.1011] Рекомендация МСЭ-Т J.1011 (2016 год), *Встроенный общий интерфейс для заменяемых решений CA/DRM; архитектура, определения и обзор*.

[ITU-T J.1013] Recommendation ITU-T J.1013 (2020), *Embedded common interface for exchangeable CA/DRM solutions; The virtual machine.*

[ITU-T J.1014] Рекомендация МСЭ-T J.1014 (2020 год), *Встроенный общий интерфейс для заменяемых решений CA/DRM; усовершенствованная система безопасности – ориентированные на ECI функциональные возможности*.

[ITU-T J.1015] Recommendation ITU-T J.1015 (2020), *Embedded common interface for exchangeable CA/DRM solutions; The advanced security system – Key ladder.*

[ITU-T T.871] Recommendation ITU-T T.871 (2011), *Information technology – Digital compression and coding of continuous-tone still images: JPEG File Interchange Format (JFIF).*

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

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

[ISO/IEC 13818-1-1] ISO/IEC 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/IEC 21320] ISO/IEC 21320, *Information technology – Document Container File – Part 1: Core.*

[IETF RFC 4122] IETF RFC 4122 (July 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/IEC 13818-6] ISO/IEC 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/IEC 7816-1] ISO/IEC 7816-1, *Identification cards – Integrated circuit cards – Part 1: Cards with contacts – Physical Characteristics.*

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

[ISO/IEC 7816-3] ISO/IEC 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/IEC 7816-5] ISO/IEC 7816-5, *Identification cards – Integrated circuit cards – Part 3: Cards with contacts – Registration of application providers.*

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

[ISO/IEC 23001-9] ISO/IEC 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/IEC 14496-12] ISO/IEC 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 (Second Edition).*

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

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

[ISO/IEC 8859-1] ISO/IEC 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 cod*e.

[ISO/IEC 62766-5-2] ISO/IEC 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/IEC 7816-4] ISO/IEC 7816-4, *Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange.*

3 Определения

3.1 Термины, определенные в других документах

Отсутствуют.

3.2 Термины, определенные в настоящей Рекомендации

В настоящей Рекомендации определяются следующие термины.

В настоящей Рекомендации термины, выделенные полужирным шрифтом и начинающиеся с заглавной буквы, имеют особые значения в контексте ECI, которые могут отличаться от общеупотребительных.

3.2.1 Усовершенствованная система безопасности (Advanced Security System – AS System) – функция устройства **CPE**, соответствующего спецификации **ECI**, обеспечивающая повышенный уровень безопасности (оборудования и программного обеспечения) для **клиента ECI**.

3.2.2 Сегмент системы AS (AS slot) – ресурсы блока усовершенствованной системы безопасности, предоставляемые **хостом ECI** исключительно **клиенту ECI**.

3.2.3 Сеанс сегмента системы AS (AS slot session) – ресурсы и вычисления в сегменте **системы AS**, связанные с дешифрованием или повторным шифрованием элемента контента.

3.2.4 Объект-брат (Brother) – другой **дочерний объект** того же **родительского объекта**.

ПРИМЕЧАНИЕ. – Родительский**, дочерний объекты** и **объекты-братья** – это объекты, управляющие **сертификатами**.

3.2.5 Сертификат (Certificate) – структура данных, определение которой приведено в разделе 5 настоящей Рекомендации, с дополнительной защищенной цифровой подписью, идентифицирующей **объект**.

ПРИМЕЧАНИЕ. – Владелец секретного ключа подписи подтверждает правильность данных (производит аутентификацию), подписывая их своим секретным ключом. Его открытый ключ может использоваться для верификации данных.

3.2.6 Цепочка сертификатов (Certificate Chain) – перечень **сертификатов**, которые аутентифицируют друг друга вплоть до **корневого списка аннулирования** включительно.

3.2.7 Подсистема обработки сертификатов (Certificate Processing Subsystem, CPS) – подсистема **хоста ECI**, обеспечивающая проверку **сертификатов** и предоставляющая дополнительную защиту от несанкционированного вмешательства.

3.2.8 Дочерний объект, дочерние объекты (Child, Children) – **объект** (объекты), относящийся к **сертификату**, подписанному (общим) **родительским объектом**.

ПРИМЕЧАНИЕ. – **Родительский, дочерний объекты** и **объекты-братья** – это объекты, управляющие **сертификатами**: данные инициализации и программное обеспечение, которое используется для запуска системы SoC оборудования **CPE**.

3.2.9 Система защиты контента (Content Protection system) – система, входящая в **экосистему ECI**, использующая криптографические методы для управления доступом к контенту и услугам.

ПРИМЕЧАНИЕ. – Этот термин часто заменяют альтернативным термином "система защиты услуг". К типичным системам такого рода относятся, например, системы условного доступа (CAS) и системы управления цифровыми правами (DRM).

3.2.10 Оборудование в помещении клиента (Customer Premises Equipment, CPE) – устройство для приема мультимедийных данных, оборудованное интерфейсом **ECI**, обеспечивающим доступ **пользователя** к цифровым мультимедийным услугам.

3.2.11 Производитель CPE (CPE manufacturer) – компания, выпускающая устройства **CPE** с поддержкой интерфейса **ECI**.

3.2.12 Встроенный общий интерфейс (Embedded CI, ECI) – архитектура и система, определяемая в документе ETSI ISG "Embedded CI", которая позволяет разрабатывать и внедрять программно-заменяемые **клиенты ECI** в комплекты оборудования в помещении клиента (**CPE**), обеспечивая тем самым совместимость устройств **CPE** с интерфейсом **ECI**.

3.2.13 Приложение ECI (ECI application) – HTML-приложение, размещаемое в **клиенте ECI** и запускаемое в специальном сеансе работы браузера в целях взаимодействия с **пользователем** и обеспечения ввода данных **пользователя** в модуль **клиента ECI**.

3.2.14 Производитель микросхем ECI (ECI Chip Manufacturer) – компания, выпускающая однокристальные микросхемы, реализующие функциональные возможности интерфейса **ECI**.

3.2.15 Клиент ECI (клиент встроенного общего интерфейса) (Embedded CI Client, ECI Client) – практическая реализация клиента CA/DRM, соответствующая спецификациям встроенного интерфейса CI.

ПРИМЕЧАНИЕ. – Данным термином обозначается модуль программного обеспечения в оборудовании **CPE**, который обеспечивает все средства для получения защищенным образом разрешений и прав потребителя, касающихся контента, который распределяется дистрибьютером контента или **оператором**. Кроме того, в нем содержаться условия, согласно которым то или иное право или разрешение может использоваться потребителем, а также ключи для расшифровки различных сообщений и контента.

3.2.16 Образ клиента ECI (ECI Client Image) – файл с программным обеспечением, таким как код виртуальной машины, и данные инициализации, требуемые **загрузчиком клиента ECI**.

3.2.17 Загрузчик клиента ECI (ECI Client Loader) – часть модуля программного обеспечения **хоста ECI**, обеспечивающая загрузку, проверку и установку нового программного обеспечения **клиента ECI** в **контейнер ECI хоста ECI**.

3.2.18 Контейнер ECI (ECI Container) – одиночная копия VM с дополнительными (вспомогательными) библиотеками и интерфейсом API **ECI**, позволяющим запускать одиночную копию **клиента ECI** на оборудовании **CPE**.

3.2.19 Экосистема ECI (ECI Ecosystem) – коммерческая эксплуатация, охватывающая **доверительный орган** и несколько платформ, а также установленное на местах оборудование **CPE**, поддерживающее интерфейс **ECI**.

3.2.20 Хост ECI (ECI Host) – аппаратная и программная система в составе **CPE**, которая охватывает функциональные возможности, связанные с **ECI**, и включает интерфейсы для связи с **клиентом ECI**.

ПРИМЕЧАНИЕ. – **Хост ECI** входит в состав встроенного программного обеспечения **CPE**.

3.2.21 Образ хоста ECI (ECI Host Image) – файл(ы), содержащий программное обеспечение и данные инициализации для среды **ECI**.

ПРИМЕЧАНИЕ 1. – Образ **хоста ECI** может состоять из нескольких файлов **образов хоста ECI**.

ПРИМЕЧАНИЕ 2. – Он может также содержать другое программное обеспечение, не создающее помех **хосту ECI** и не допускающее возможность несанкционированного слежения за ним.

3.2.22 Загрузчик хоста ECI (ECI Host Loader) – программный модуль, позволяющий осуществлять загрузку, проверку и установку программного обеспечения **хоста ECI** в оборудование **CPE**.

ПРИМЕЧАНИЕ. – В конфигурации многоступенчатой загрузки данным термином обозначаются все критичные с точки зрения безопасности функции загрузки, задействованные при загрузке **хоста ECI**.

3.2.23 Корневой сертификат ECI (ECI Root Certificate) – **сертификат**, который выпускается для проверки элементов, одобренных **доверительным органом ECI**.

3.2.24 Объект (Entity) – организация (например, производитель, **оператор** или **поставщик систем безопасности**) или реально существующий элемент (например, **хост ECI**, **система управления платформой** или **клиент ECI**), имеющий уникальный идентификатор в **экосистеме ECI**.

3.2.25 Цепочка экспорта (Export Chain) – цепочка **сертификатов**, используемая для авторизации экспорта в одну из **микросистем DRM** или в группу микросистем.

3.2.26 Соединение экспорта (Export Connection) – удостоверенная связь между **клиентом ECI**, способным дешифровать контент, и **микросервером**, способным повторно зашифровать контент.

3.2.27 Группа экспорта (Export Group) – группа **микросистем DRM**, в которые разрешен экспорт.

3.2.28 Родительский объект (Father) – сторона, подписавшая **сертификат дочернего объекта**

ПРИМЕЧАНИЕ. – **Родительский, дочерний объекты и объект-брат** – это объекты, управляющие сертификатами.

3.2.29 Серия образов (Image Series) – серия образов для **хоста ECI** или **клиента ECI**, различающихся в зависимости от **CPE**_id оборудования **CPE**, но при этом обеспечивающих (почти) идентичные функциональные возможности.

3.2.30 Цепочка импорта (Import Chain) – цепочка от PORK **клиента ECI** до **объекта**, которая представляет собой систему экспорта или **группу экспорта**.

ПРИМЕЧАНИЕ. – **Цепочка экспорта** и соответствующая **цепочка импорта** может использоваться для аутентификации сеанса **микросервера,** импортирующего контент к экспортирующему **клиенту ECI**.

3.2.31 Соединение импорта (Import Connection) – проверенное соединение **клиента ECI** с **микросервером,** разрешающее ему импортировать дешифрованный контент для последующего повторного шифрования.

3.2.32 Производитель (Manufacturer) – объект, разрабатывающий и реализующий оборудование CPE, которое обеспечивает внедрение системы **ECI** и позволяет устанавливать **хост ECI** и модули **клиентов ECI** посредством загрузки программного обеспечения.

3.2.33 Указатель медиаданных (Media Handle) – ссылка на отдельный набор программных установок для дешифрования или повторного шифрования между **клиентом ECI** и **хостом ECI**.

3.2.34 Микроклиент (Micro Client) – **клиент ECI** или клиент, не совместимый с **ECI**, способный расшифровать контент, повторно зашифрованный **микросервером**.

3.2.35 Микросервер (Micro Server) – **клиент ECI**, способный импортировать дешифрованный контент, повторно шифровать его и аутентифицировать определенный **клиент ECI** или группу **клиентов ECI** как **целевой объект** для последующего дешифрования.

3.2.36 Микросистема DRM (Micro DRM System) – **система защиты контента**, повторно шифрующая контент на оборудовании **CPE** с помощью **микросервера** и позволяющая расшифровывать повторно зашифрованный контент посредством аутентифицированных **микроклиентов**.

ПРИМЕЧАНИЕ. – **Микросервер** и **микроклиенты** предоставляются оператором **микросистемы DRM**.

3.2.37 Оператор (Operator) – организация, выполняющая роль **системы управления платформой** и обладающая правом подписи в **экосистеме ECI**, выданным **доверительным органом ECI**.

ПРИМЕЧАНИЕ. – **Оператор** может осуществлять управление несколькими **платформами**.

3.2.38 Система управления платформой (Platform Operation, PO) – конкретный пример процесса оказания технической услуги, обладающего едиными идентификационными данными **ECI** с точки зрения безопасности.

3.2.39 Сеанс повторного шифрования (Re-encryption Session) – управляемый **микросервером** процесс, включающий в себя импорт контента из **соединения импорта**, его повторное шифрование и выдачу информации для последующего ее дешифрования аутентифицированным **целевым объектом**.

3.2.40 Запрос (Request) – сообщение от отправителя для получателя с требованием предоставления определенной информации либо выполнения определенной процедуры в рамках **экосистемы ECI**. Параметры запроса указываются в полях данных этого запроса.

ПРИМЕЧАНИЕ. – Подробная информация приведена в пункте 9.2.3.

3.2.41 Отклик (Response) – сообщение, отправленное в рамках **экосистемы ECI** в ответ на **запрос**.

ПРИМЕЧАНИЕ. – Подробная информация приведена в пункте 9.2.3.

3.2.42 Список аннулирования (Revocation List, RL) – список **сертификатов,** которые были аннулированы и поэтому больше не должны использоваться.

3.2.43 Корневой элемент (Root) – открытый ключ или **сертификат**, содержащий открытый ключ. **Корневой элемент** является основой для аутентификации цепочки **сертификатов**.

3.2.44 Защищенный аутентифицированный канал (Secure Authenticated Channel, SAC) – канал связи, установленный между двумя **объектами**, в котором **объекты** провели защищенную идентификацию друг друга (аутентификацию) и условились о шифровании передаваемых между ними данных (в защищенном режиме).

3.2.45 Услуга (Service) – контент, передаваемый **системой управления платформой**.

ПРИМЕЧАНИЕ. – В контексте **ECI** рассматривается только защищенный контент.

3.2.46 Открытый ключ отправителя (Sender Public Key, SPK) – открытый ключ отправителя зашифрованного контента, который используется в **экосистеме ECI** для проверки происхождения подписи первого ключа цепочки, применяемой для дешифрования контента, причем отправитель является частью системы управления платформой.

3.2.47 Смарт-карта (Smart Card) – съемное защищенное устройство, используемое несколькими поставщиками услуг CA или DRM в целях повышения уровня безопасности своих продуктов в **экосистеме ECI**.

3.2.48 Целевой объект (Target) – **микроклиент** или группа **микроклиентов**, для которых контент повторно шифруется **микросервером**.

3.2.49 Доверительный орган (Trust Authority, TA) – организация, регулирующая все правила и положения, применяемые при конкретной реализации **ECI**, на которую ориентируются на определенном рынке.

ПРИМЕЧАНИЕ. – **Доверительный орган** должен являться юридическим лицом, который может добиваться выполнения правовых требований. **Доверительный орган** должен быть беспристрастным по отношению ко всем участникам **экосистемы ECI**, которую он регулирует.

3.2.50 Доверенная третья сторона (Trusted Third Party, TTP) – поставщик услуг по вопросам безопасности, который выдает **сертификаты** и ключи **производителям** компонентов системы, соответствующих спецификациям **ECI-систем**.

ПРИМЕЧАНИЕ. – Доверенная третья сторона находится под контролем **доверительного органа (TA)**.

3.2.51 Пользователь (User) – лицо, эксплуатирующее устройство, соответствующее спецификациям **ECI**.

3.2.52 Копия виртуальной машины (VM Instance) – создаваемый хостом **ECI** элемент виртуальной машины, который представляется как среда для работы **клиента ECI**.

4 Сокращения и акронимы

В настоящей Рекомендации используются следующие сокращения и акронимы:

4CC	Four Character Code (FourCC)	Четырехсимвольный код
3DES	Triple-DES	Стандарт трехкратного шифрования данных
AEAD	Authenticated Encryption with Associated Data	Аутентифицированное шифрование с присоединенными данными
AES	Advanced Encryption standard	Усовершенствованный стандарт шифрования

AES-GCM	AES Galois Counter mode	Режим счетчика Галуа с AES-шифрованием
AID	Application Identifier	Идентификатор приложения
AK	Authentification Key	Ключ аутентификации
APDU	Application Protocol Data Unit	Блок данных прикладного протокола
API	Application Programming Interface	Интерфейс прикладного программирования
AS	Advanced Security	Усовершенствованная система безопасности
ASCII	American Standard Code for Information Interchange	Американский стандартный код для обмена информацией
ATR	Answer to Reset	Ответ для сброса
BAT	Bouquet Association Table	Таблица объединения букета программ
BMFF	Base Media File Format	Основной формат файлов мультимедиа
BSD	Berkeley Software Distribution	Система распространения программного обеспечения Беркли
CA	Conditional Access	Условный доступ
CA/DRM	Conditional Access/Digital Rights Management	Условный доступ/управление цифровыми правами
CAT	Conditional Access Table	Таблица условного доступа
CBC	Cipher Block Chaining	Цепочка блоков шифра
CENC	Common Encryption	Общее шифрование
CI	Common Interface	Общий интерфейс
CP	Content Property	Свойство контента
CPE	Customer Premises Equipment	Оборудование в помещении клиента
CPS	Certificate Processing Subsystem	Подсистема обработки сертификатов
CPU	Central Processing Unit	Центральный процессор
CRC	Cyclic Redundancy Check	Циклическая проверка избыточности
CRL	Certificate Revocation List	Список аннулирования сертификатов
CSA	Common Scrambling Algorithm	Общий алгоритм скремблирования
CSA1	Common Scrambling Algorithm, first version	Общий алгоритм скремблирования, первая версия
CSA3	Common Scrambling Algorithm, third version	Общий алгоритм скремблирования, третья версия
CSS	W3C Cascading Style Sheets	Каскадные таблицы стилей W3C
CSS3	CSS version 3	CSS версия 3
CTR	Counter Mode	Режим счетчика
CW	Control Word	Контрольное слово
Dash	Dynamic Adaptive Streaming over HTTP	Динамическая адаптивная потоковая передача данных по протоколу HTTP
DDB	Download Data Block	Блок загружаемых данных
DDOS	Distributed Denial of Service	Распределенная атака типа "отказ в обслуживании"
DES	Data Encryption Standard	Стандарт шифрования данных

DHE	Ephemeral Diffie-Hellman	Эфемерный ключ Диффи–Хеллмана
DII	Download Info Indication	Отображение информации о загрузке
DLNA	Digital Living Network Alliance	Альянс цифровой домашней сети
DNS	Domain Name System	Система наименований доменов
DRM	Digital Rights Management	Система управления цифровыми правами
DSI	Download Server Initiate	Инициация сервера загрузки
DSMCC	Digital Storage Media Command and Control	Система команд и управления для средств хранения цифровой информации
DVB	Digital Video Broadcasting	Цифровое телевизионное вещание
EAC	Export Authorization Certificate	Сертификат авторизации экспорта
EAOC	Export Authorization Operator Certificate	Операторский сертификат авторизации экспорта
ECM	Entitlement Control Message	Сообщение для контроля прав доступа
EGC	Export Group Certificate	Сертификат группы экспорта
EIT	Event Information Table	Таблица информации о событиях
EMM	Entitlement Management Message	Сообщение для управления правами доступа
ES	Elementary Stream	Элементарный поток
ESC	Export System Certificate	Сертификат системы экспорта
GCM	Galois/Counter Mode	Режим счетчика Галуа
GMT	Greenwich Mean Time	Среднее гринвичское время
HD	High Definition	Высокая четкость
HDCP	High-bandwidth Digital Content Protection	Защита цифрового широкополосного контента
HTML	Hyper Text Mark-up Language	Язык разметки гипертекстовых документов
HTTP	HyperText Transfer Protocol	Протокол передачи гипертекста
HTTP(S)	Hypertext Transfer Protocol Secure	Защищенный гипертекстовый транспортный протокол
iDTV	integrated Digital TV receiver	Интегрированный цифровой телевизионный приемник
IFSC	Information Field Size of Card	Размер информационного поля карты
IFSD	Information Field Size of Device	Размер информационного поля устройства
IP	Internet Protocol	Протокол Интернет
IPTV	TV using the Internet Protocol (IP)	Телевидение на основе протокола Интернет (IP)
IPv4	Internet Protocol Version 4	Протокол Интернет версии 4
IPv6	Internet Protocol Version 6	Протокол Интернет версии 6
ISO	International Organization for Standardisation	Международная организация по стандартизации (ИСО)
ISOBMFF	ISO Base Media File Format	Основной формат файлов мультимедиа ИСО
LAN	Local Area Network	Локальная сеть
LSB	Least Significant Bit	Младший двоичный разряд
MIME	Multipurpose Internet Mail Extensions	Многоцелевые расширения электронной почты в интернете

MMI	Man Machine Interface	Интерфейс "человек–машина"
MP4	Digital Multimedia Container Format (asl called MPEG4 part 14)	Формат цифрового мультимедийного контейнера (также называемый MPEG4, часть 14)
MPD	Media Presentation Description	Описание представления мультимедийных данных
MPEG	Motion Picture Experts Group	Экспертная группа по движущимся изображениям
MSB	Most Significant Bit	Старший двоичный разряд
n.a.	not applicable	не применимо
NV memory	Non-Volatile memory	Энергонезависимая память
NV	Non-Volatile	Энергонезависимый
OS	Operating System	Операционная система
OTT	Over The Top (over the open Internet)	Технология OTT (доставка поверх открытого интернета)
OUI	Organizationnally Unique Identifier	Уникальный идентификатор организации
PAT	Program Association Table	Таблица ассоциаций программ
PayTV	Pay Television	Платное телевидение
PES	Packet Elementary Stream	Элементарный поток пакетов
PID	MPEG Packet Identifier	Идентификатор пакетов MPEG
PIN	Personal Identification Number	Персональный идентификационный номер
PKIX	Public-Key Infrastructure X.509	Инфраструктура открытого ключа X.509
PMT	Program Map Table	Таблица программного плана
PO	Platform Operation	Система управления платформой
POC	Platform Operation Certificate	Сертификат системы управления платформой
POPK	Platform Operation Public Key	Открытый ключ системы управления платформой
PPS	Protocol and Parameter Selection	Выбор протоколов и параметров
PSI	Program Specific Information	Специальная информация о программе
PSSH	Protection System Specific Header	Специальный заголовок системы защиты
PVR	Personal Video Recorder	Персональный видеомагнитофон
RAM	Random Access Memory	Оперативное запоминающее устройство
RFU	Reserved for Future Use	Зарезервировано для использования в будущем
RL	Revocation List	Список аннулирования
SAC	Secure Authenticated Channel	Защищенный аутентифицированный канал
SDT	Service Description Table	Таблица с описанием услуг
SHA	Secure Hash Algorithm	Защищенный алгоритм хеширования
SI	Service Information	Служебная информация
SIM	Subscriber Identity Module	Модуль идентификации абонента
SoC	System on Chip	Однокристальная система

SPK	Signature Public Key (also known as Signature Verification Key)	Открытый ключ проверки подписи
SSK	Signature Secret Key (also known as Signature Private Key)	Закрытый ключ проверки подписи
SSL	Secure Sockets Layer	Уровень защищенных разъемов
SSU	System Software Update	Обновление системного ПО
STB	Set Top Box	Абонентская приставка
TA	Trust Authority	Доверительный орган
TCK	The CheckK byte	Контрольный байт
TCP	Transmission Control Protocol	Протокол управления передачей
TLS	Transport Layer Security	Безопасность транспортного уровня
TPC	Transmission Control Protocol	Протокол управления передачей
TPDU	Transport Protocol Data Unit	Блок данных протокола передачи
TPEGC	Third Party Export Group Certificate	Сертификат группы экспорта третьей стороны
TS	Transport Stream	Транспортный поток
TTP	Trusted Third Party	Доверенная третья сторона
TV	Television	Телевидение
UDP	User Datagram Protocol	Протокол датаграмм пользователя
UHD	Ultra High Definition	Сверхвысокая четкость
UI	User Interface	Пользовательский интерфейс
uimsbf	unsigned integer, most significant bit first	Целое без знака, старший двоичный разряд
UNT	Update Notification Table	Таблица уведомлений об обновлениях
URI	Usage Rights Information	Информация о правах на использование
URL	Uniform Resource Locator	Унифицированный указатель ресурсов
USB	Universal Serial Bus	Универсальная последовательная шина
UTF	UCS (Universal Character Set) Transformation Format	Формат преобразования UCS (универсального набора символов)
UUID	Universally Unique Identifier	Универсальный уникальный идентификатор
VM	Virtual Machine	Виртуальная машина
WAN	Wide Area Network	Территориальная распределенная сеть
WEB	World Wide Web	Всемирная сеть связи (Всемирная паутина)

5 Система сертификатов ECI

5.1 Введение

5.1.1 Сфера применения

В интерфейсе **ECI сертификаты** используются для различных целей, таких как **загрузчик хоста ECI**, **загрузчик клиента ECI** и **сертификаты операторов ECI**. В настоящем разделе приводится определение этих **сертификатов** и соответствующего **списка аннулирования**, а также описывается их построение в цепочки и структура **корневого сертификата**. В этом определении используется

описываемый в настоящей Рекомендации компактный двоичный формат, пригодный для аппаратной реализации и для криптографии, а также простые системы сигнализации для будущих версий и расширений.

5.1.2 Нотация и условные обозначения полей

Приведенные ниже определения структуры данных отображаются непосредственно на последовательности байтов. Любая криптографическая функция определяется для работы с конкретным представлением последовательности байтов.

Определение данных соответствует естественному выравниванию 16- и 32-байтовых полей, что упрощает обработку данных на 32-битовом процессоре. Заполнение используется в качестве общего поля для указания обязательных для заполнения полей. При этом используется функция padding(n_bytes), где n_bytes является границей выравнивания, выраженная в количестве байтов от начала определенной структуры данных. Поля заполнения должны пропускаться при интерпретации структур данных. Значение поля заполнения устанавливается равным 0.

Любое поле, определяемое с помощью другой структуры данных через определение типа, не имеет мнемоники. Как правило, для такого поля не задается определение длины поля.

5.1.3 Поле расширения

Многие из определяемых значимых структур данных имеют поле расширения, которое позволяет добавлять будущие (обратно совместимые) расширения. Определение представлено в таблице 5.1.3-1.

Таблица 5.1.3-1 – Определение поля расширения

Синтаксис	Количество битов	Мнемоника
Extension_Field {		
padding(4)		
length	32	uimsbf
для (i=0; i<length; i++) {		
extension_byte	8	uimsbf
}		
}		

Семантика

length: integer	Количество байтов в цикле, следующим за данным полем. Значение должно быть кратно 4 и может равняться нулю
extension_byte: byte	Поле данных, содержащее информацию, которая может игнорироваться действующими системами, основанными на версиях настоящего документа, в которых содержимое данного поля не определено

5.2 Сертификаты ECI

Сертификат ECI имеет простую структуру. В отличие от используемых в интернете сертификатов X.509, идентификатором **сертификата** является простое двоичное число, предназначенное только для машинных интерпретаций.

Общая структура **сертификата** изображена на рисунке 5.2-1.

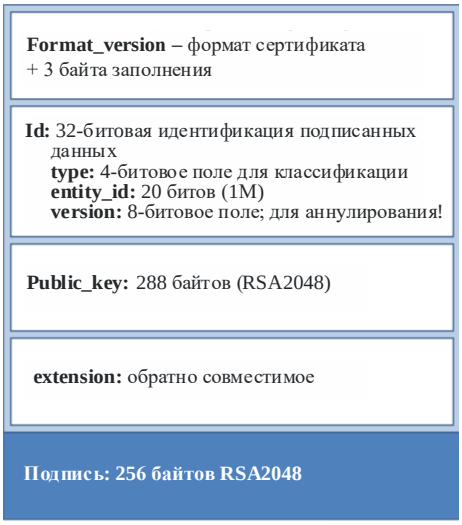


Рисунок 5.2-1 – Формат сертификата ECI, версия 1

Формат **сертификата ECI** определяется в таблице 5.2-1.

Каждый подписанный элемент использует отдельное 8-байтовое стартовое поле, где первый байт является форматом версии подписанного элемента, затем (для элементов версии 1) следуют 3 байта заполнения, а далее – вторая группа из 4 байтов, представляющих уникальный идентификатор в контексте секретного ключа подписывающего объекта.

Таблица 5.2-1 – Определение сертификата ECI

Синтаксис	Количество битов	Мнемоника
ECI_Certificate_Id {		
padding(4)		
Type	4	uimsbf
entity_id	20	uimsbf
Version	8	uimsbf
}		
ECI_Public_Key_v1 {		
byte **modulus**[256]	2048	
}		
ECI_Certificate_Data_v1 {		
ECI_Certificate_Id **id**	32	uimsbf
Public_Key_v1 **public_key**	2304	
Extension_Field **extension**		
}		
ECI_Signature_v1 {		
byte **signature**[256]	2048	uimsbf
}		
ECI_Certificate {		
format_version	8	uimsbf
если (version == 0x01) {		
ECI_Certificate_Data_v1 **data**		
ECI_Signature_v1 **signature**		
}		
}		

Семантика

format_version: integer	Значения 0x00, 0x02..0xFF зарезервированы. Значение 0x01: формат **сертификата ECI**, версия 1. Действующие системы, которые не распознают тип **сертификата**, не обрабатывают его и в ответ на запросы о подтверждении отправляют сообщение об ошибке
id: integer	Идентификация сертификата в виде 32-битового числа, которое является уникальным в контексте **родительского объекта сертификата** (подписавшего сертификат). Значения 0x00000 and 0xF0000-0xFFFFF зарезервированы
type: integer	Поле **type** определяет тип объекта, например, **производитель, хост ECI, оператор** и т. д. в контексте объекта, подписавшего сертификат (**родительского объекта**). **Сертификаты**, имеющие значение типа 0x0.. 0x7 требуют **список аннулирования** для проверки **дочерних объектов**. Значения типа 0x8 и выше не должны требовать **список аннулирования** для проверки **дочерних объектов** (см. таблицу 5.2-2)
entity_id: integer	Определяет номер объекта. **Идентификатор entity_id** заключает в себе различные субформаты согласно типу **сертификата**. Если не указано иное, значения entity_id уникальны в контексте **родительского объекта** (подписавшего **сертификат** или **список аннулирования**)
version	Номера версии сертификатов объектов, расположенные в порядке возрастания (как правило, с шагом 1)
extension: Extension_Field	Данные в этом поле игнорируются функциями обработки данных, которые не должны их интерпретировать. Указанное поле может использоваться для специальных данных в случаях конкретного применения общего определения **сертификата**. Это поле интерпретируется в зависимости от контекста и оно не должно использоваться приложениями, не поддерживающими **ECI**, если только на этот счет нет особого разрешения
public_key: ECI_Public_Key_v1	Открытый ключ (присвоенный **родительским объектом**) объекта данного **сертификата**
data: ECI_Certificate_Data	Это раздел данных **сертификата**
signature: byte[256]	Поле подписи содержит представление последовательности байтов подписи **родительского объекта сертификата** с применением криптографических функций согласно определению, приведенному в приложении A

Любая проверка **сертификата ECI** включает проверку общей длины **сертификата** с точки зрения накопления определений поля.

Для большинства **сертификатов** и **списков аннулирования** используются значения обобщенного типа, которые гарантируют уникальность всех присвоенных значений. Обзор всех данных, подписанных **доверительным органом ECI**, представлен в таблице 5.2-2.

Таблица 5.2-2 – Присвоение идентификаторов и родительские объекты для подписанных элементов

Родительский объект	Тип	Поле идентификатора	Описание
Корневой сертификат	0x0	0xFFFFF	Корневой сертификат
Корневой сертификат	0x1	Manufacturer id,<> 0xFxxxx	**Сертификат** производителя
Корневой сертификат	0x1	Manufacturer RL id, == 0xFxxxx	**Список аннулирования** производителя
Производитель	0x0	Host_id, <> 0xFxxxx	**Сертификат хоста ECI**
Производитель	0x0	Host RL, == 0xFxxxx	**Список аннулирования хоста ECI**
Хост	0x8	Host Image id	**Образ хоста ECI**
Хост	0x9	Host Image Series id	**Сертификат серии образов хоста ECI**
Серия образов хоста	0x9	Image Target Id	Образ серии **хоста ECI**
Корневой сертификат	0x2	Vendor id, <> 0xFxxxx	**Сертификат поставщика по обеспечению безопасности**
Корневой сертификат	0x2	Vendor RL id, == 0xFxxxx	**Список аннулирования сертификатов поставщика по обеспечению безопасности**
Поставщик	0x0	Client id, <> 0xFxxxx	**Сертификат клиента ECI**
Поставщик	0x0	Client RL, == 0xFxxxx	**Клиент ECI** и **список аннулирования** серии **клиентов ECI**
Клиент	0x0	Client id	**Образ клиента ECI**
Клиент	0x1	Client series id	**Сертификат** серии клиентов
Серия клиентов	0x8	Image Target Id	Образ серии клиентов
Корневой сертификат	0x3	Operator id, <> 0xFxxxx	**Сертификат** оператора
Корневой сертификат	0x3	Operator RL id, == 0xFxxxx	**Список аннулирования** оператора
Оператор	0x0	Platform Operation id, <> 0xFxxxx	**Сертификат системы управления платформой**

Таблица 5.2-2 – Присвоение идентификаторов и родительские объекты для подписанных элементов

Родительский объект	Тип	Поле идентификатора	Описание
Оператор	0x0	Platform Operation RL id, == 0xFxxxx	Список аннулирования **системы управления платформой**
Система управления платформой	0x0	Platform Operation Client Image Cosignature id <> 0xFxxxx	Совместная подпись образа клиента **системы управления платформой**
Система управления платформой	0x0	Platform Operation Client Image RL id == 0xFxxxx	**Список аннулирования образа клиента системы управления платформой**
Система управления платформой или целевая группа	0x0	Target Group id, <> 0xFxxxx	Целевая группа, определенная в [ITU-T J.1014]
Система управления платформой или целевая группа	0x0	Target RL id, == 0xFxxxx	Список аннулирования целевых объектов, определенный в [ITU-T J.1014]
Система управления платформой или целевая группа	0x8	Micro Client id, <> 0xFxxxx	Микроклиент, определенный в [ITU-T J.1014]
Система управления платформой, **группа экспорта**, группа экспорта третьей стороны	0x4	Export Group id, <> 0xFxxxx	**Группа экспорта**
Система управления платформой, **группа экспорта**, группа экспорта третьей стороны	0x4	Export Group RL id, ==0xFxxxx	Список аннулирования **группы экспорта**
Группа экспорта	0x5	Third Party Export Group id, <> 0xFxxxx	**Группа экспорта** третьей стороны
Группа экспорта	0x8	Export Group RL id, == 0xFxxxx	Список аннулирования **группы экспорта**
Группа экспорта, группа экспорта третьей стороны	0xE	Export System id, <> 0xFxxxx	Система экспорта
Корневой сертификат	0x4	Export Authorization Operator id, <> 0xFxxxx	Оператор авторизации экспорта
Корневой сертификат	0x4	Export Authorization Operator id, == 0xFxxxx	Список аннулирования оператора авторизации экспорта
Оператор авторизации экспорта, авторизация экспорта	0x0	Export Authorization id, <> 0xFxxxx	Авторизация экспорта (с **дочерними объектами**)
Оператор авторизации экспорта, авторизация экспорта	0x0	Export Authorization id, == 0xFxxxx	Список аннулирования авторизации экспорта
Прочие	Прочие		Зарезервировано
ПРИМЕЧАНИЕ. – Функции ECI могут транспортировать и обрабатывать поле **данных** и разделы **подписи сертификата**, а также прочие подписанные элементы данных по отдельности.			

5.3 Список аннулирования ECI

Объект, изначально подписавший аннулируемый **сертификат**, подписывает также **список аннулирования**. **Список аннулирования** представляет собой список идентификаторов объектов, определяющий минимальную допустимую версию для их **сертификатов**. Номер минимальной версии **списка аннулирования** применим для **сертификата**, являющегося записью в **списке аннулирования** и имеющего соответствующий список (списки) аннулирования. Структура **списка аннулирования ECI** определяется на рисунке 5.3-1.

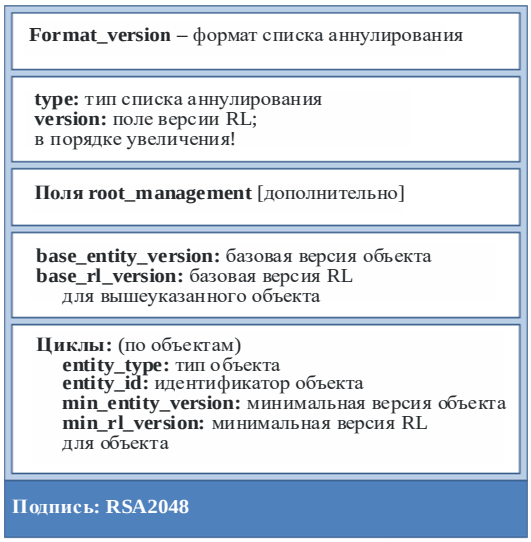


Рисунок 5.3-1 – Структура списка аннулирования

Введенные в действие **хосты ECI** хранят последний (определяемый в **rl_version**) **список аннулирования**, полученный для **объекта**, которым они управляют, независимо от источника данных.

Список аннулирования (ECI_RL) определяется в таблице 5.3-1.

Таблица 5.3-1 – Определение списка аннулирования

Синтаксис	Количество битов	Мнемоника
ECI_RL_Id {		
padding(4)		
Тип	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
для (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
если (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**	2048 (см. примечание)	uimsbf
}		
}		
ПРИМЕЧАНИЕ. – = в списках аннулирования, относящихся к версии 1 **сертификата**.		

Семантика

format_version: integer	Значения 0x00, 0x02..0xFF зарезервированы. Значение 0x01: формат **списка аннулирования** ECI, версия 1. Действующие системы, которые не распознают тип **сертификата**, не обрабатывают его и в ответ на запросы о подтверждении отправляют сообщение об ошибке
type: integer	Тип поля определяется в ECI_Certificate_Id, см. таблицу 5.3-1
indicator: integer	Индикация **списка аннулирования**; значение должно быть равным 0xF
version: integer	Версия данного списка аннулирования. Начинается с 1 (если **сертификат** новый, поле, как правило, пустое) и увеличивается при каждом обновлении
base_entity_version: integer	Все объекты, у которых **id.version** меньше чем **base_id_version**, аннулируются
base_rl_version	Все списки аннулирования для объекта с сертификатом версии **base_entity_version**, которая меньше чем **base_rl_version**, считаются недействительными
number_of_entities: integer	Количество объектов в списке аннулирования. Максимальные значения см. в таблице 5.3-1
entity_type: integer	Тип объекта, более ранние версии сертификатов которого аннулируются
entity_id: integer	Entity_id объекта, более ранние версии сертификатов которого аннулируются
min_entity_version: integer	Номер минимальной версии объекта (id сертификата), соответствующий **entity_type** и **entity_id**. Более ранние версии аннулируются
min_rl_version	Минимальная версия списка аннулирования, применимая в сочетании с объектом, соответствующим **entity_type**, **entity_id** и **entity_min_version**. Более ранние версии списка аннулирования считаются недействительными
root_version_indicator: bit	Если значение равно 0, поля root_version и min_root_version являются несущественными. Если значение равно 1, а **родительский объект** является **корневым сертификатом**, поля **root_version** и **min_root_version** интерпретируются как указано ниже
root_version	Версия **корневого сертификата**, подписавшего данный **лист аннулирования**
min_root_version: integer	Если версия **родительского объекта** (то есть корневого сертификата) больше или равна данному полю, все версии **корневых сертификатов**, меньшие чем **min_root_version**, аннулируются в целях проверки **сертификатов**, тип которых определяется в **revocation_id_lead**
extension: Extension_Field	Дополнительные данные: игнорируются (за исключением вычисления подписи) действующими системами, которые не рассчитаны на интерпретацию данного поля, за исключением вычисления подписи
rl_signature: ECI_Signature_v1	Подпись **объекта ECI**, с которым связан **список аннулирования**. Подпись вычисляется на основе всех предшествующих данных

ПРИМЕЧАНИЕ. – Введенное в эксплуатацию аппаратное обеспечение способно обрабатывать **списки аннулирования** по частям, выполняя поиск идентификатора последующего **сертификата**, который должен проходить валидацию при суммировании хеша подписи, а также в конце списка, где происходит проверка подписи.

Согласно общему правилу, **хосты ECI** хранят **списки аннулирования доверительного органа** для всех **сертификатов**, которые необходимы для проверки объектов, загружаемых **хостом ECI**. **Хосты ECI** заменяют сохраненный **список аннулирования** для того или иного **сертификата** или элемента вновь полученным **списком аннулирования** с номером последней версии.

Максимальная длина **списков аннулирования** должна соответствовать указаниям, приведенным в разделе B.2.

5.4 Цепочки сертификатов и деревья списков аннулирования

5.4.1 Определения структуры данных

Цепочка сертификатов представляет собой последовательность **сертификатов** с соответствующими **списками аннулирования**, в которой каждый **сертификат** подписывается объектом, управляющим предыдущим **сертификатом**. Цепочка сертификатов начинается со **списка аннулирования родительского сертификата** (как правило, это корневой сертификат). Номер минимальной (действующей) версии **сертификата** и минимальная (действующая) версия **списка аннулирования дочернего объекта** определяются **списком аннулирования** его **родительского объекта**. Цепочки применяются в качестве идентификационных данных для проверки элемента, подлежащего загрузке. Таким образом, **сертификат** обычно не отображается в **списке аннулирования** предыдущего сертификата. Тем не менее обработка **списка аннулирования** обязательна в целях проверки целостности цепочки. В таблице 5.4-1 представлена структура типичной **цепочки сертификатов**.

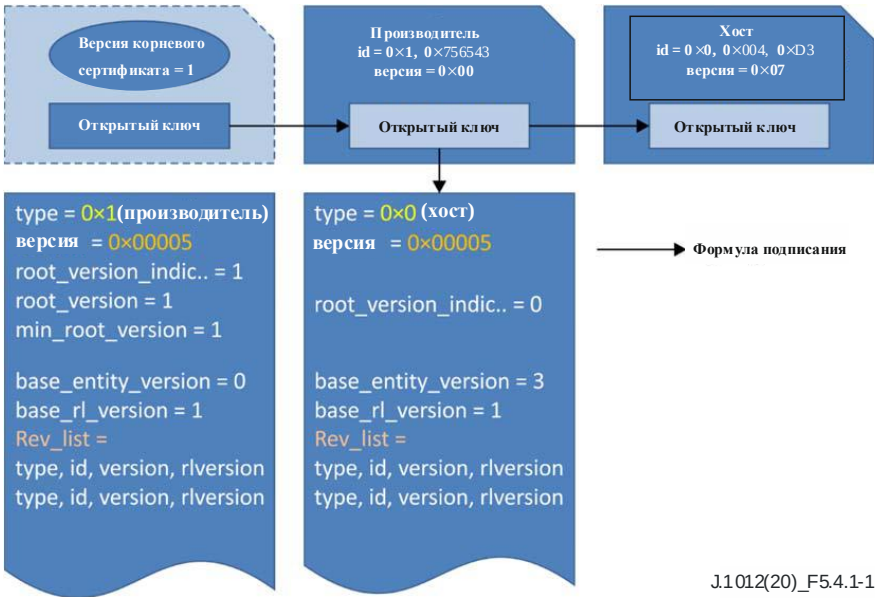


Рисунок 5.4.1-1 – Пример цепочки сертификата хоста

Цепочки могут перемещаться или храниться, а также могут состоять из разных секций.

Деревья списков аннулирования представляют собой последовательности связанных списков аннулирования, которые используют **сертификат** в предыдущей цепочке в качестве **родительского объекта**. Таким образом охватывается значительное количество сертифицированных элементов. Деревья списков аннулирования могут использоваться **системами управления платформой** в целях исключения (указания об аннулировании) других объектов, аннулированных **доверительным органом**. Определение **цепочки сертификатов** и **дерева списка** аннулирования должно выполняться в соответствии с таблицей 5.4.1-1.

Таблица 5.4.1-1 – Определения цепочки сертификатов и дерева списка аннулирования

Синтаксис	Количество битов	Мнемоника
ECI_Certificate_Chain {		
chain_length	8	uimsbf
padding(4)		
для (i=0; i<**chain_length**; i++){		
ECI_RL **rl**		
ECI_Certificate **certificate**		
}		
}		
ECI_RL_Tree {		
ECI_RL **father_revocation_list**		
three_breadth	32	uimsbf
для (i=0; i<**three_breadth**; i++){		
father_node_depth	8	uimsbf
chain_length	8	uimsbf
padding(4)	16	uimsbf
для (i=0; i<**chain_length-1**; i++){		
ECI_Certificate **certificate**		
ECI_RL **rl**		
}		
}		
}		

Семантика

chain_length: integer	Длина цепочки
rl: ECI_RL	**Список аннулирования** для предшествующего **сертификата** или **родительского объекта** в цепочке при первой итерации цепочки. Номера версий поля идентификатора **списков аннулирования** в цепочки должны быть равны
certificate: ECI_Certificate	**Родительский объект** следующего **сертификата** в текущей последовательности
father_revocation_list: ECI_RL	**Список аннулирования** для **родительского сертификата** цепочки
three_breadth: integer	Количество субцепочек в дереве
father_node_depth: integer	Уровень **родительского сертификата** в предшествующей **цепочке сертификатов** (включая **родительский объект** дерева). Унаследованный список **родительского объекта** является **родительским объектом** данной цепочки, которому предшествует его собственный **родительский объект;** и т. д. до **родительского объекта** самого дерева

Правила упорядочения **сертификатов** в **деревьях списка** аннулирования:

- деревья не содержат дубликаты **сертификатов**;
- конкретное дерево упорядочивается таким образом, что все **объекты-братья сертификата** последнего листа перечисляются как поддеревья chain_length = 0 непосредственно за последним **сертификатом**. Затем следуют поддеревья **объекта-брата**, относящегося к **родительскому объекту**, и т. д;
- сертификаты **объектов-братьев** отображаются в порядке возрастания идентификаторов в дереве (первым отображается наименьший).

5.4.2 Правила обработки цепочек сертификатов

Хост ECI выполняет проверку **цепочек сертификатов** и предоставляет соответствующий ответ для аннулируемых элементов, используя **усовершенствованную систему безопасности**. **Усовершенствованная система безопасности** выполняет ключевые этапы проверки безопасности **сертификатов** и **списка аннулирования**. Кроме того, **усовершенствованная система безопасности** позволяет **клиентам ECI** последовательно проверять правильность номеров версий цепочек, применяемых для аннулирования.

Хост ECI способен обрабатывать **цепочку сертификатов** посредством итерации. Обработка начинается с корневого **списка аннулирования доверительного органа ECI** и заканчивается последним элементом в цепочке. Обработка **цепочки сертификатов** нарушается при любом сбое промежуточной проверки. В случае сбоя **хост EC**I гарантирует, что перед введением мер согласно политике **хоста ECI**, применимых к аннулируемым объектам или недействительным идентификационным данным, текущий сертификат и **список аннулирования**, а также все предшествующие **списки аннулирования** и **сертификаты** подтверждаются соответствующими подписями. **Усовершенствованная система безопасности, определенная в** [ITU-T J.1014] и [ITU-T J.1015], гарантирует, что обработка **цепочки сертификатов** поддерживает соответствующий уровень устойчивости.

В течение периода, на протяжении которого обработка обеспечивает одинаковый результат с точки зрения приемлемости цепочек, разрешен любой порядок обработки.

1) **Хост ECI** выполняет следующие этапы проверки **списков аннулирования**.

   a) Хост **ECI** проверяет поле **format_version списка аннулирования** на соответствие версии, которую он способен интерпретировать, и обеспечивает соответствие полей **rl_id.type** и **rl_id.rl_indicator** расчетным значениям.

   b) **Хост EC**I проверяет, соответствует ли длина **списка аннулирования** значениям его полей.

   c) Если поле **root_version_indicator**=1, **хост ECI** проверяет, будет ли корневой сертификат предполагаемым **родительским объектом** на данном этапе обработки цепочки, представлено ли поле **root_version** для проверки и не превышает ли поле **min_root_version** какую-либо корневую версию, используемую в процессе обработки цепочки в данный момент.

   d) **Хост** ECI проверяет, не является ли данный **список аннулирования** недействительным по номеру минимальной версии данного **списка аннулирования**, полученному

от предыдущего **списка аннулирования** в цепочке, либо не является ли корневой список аннулирования недействительным по номеру min_root_revocation_list, используемому в данный момент в процессе обработки цепочки.

e) **Хост** ECI проверяет подпись **списка аннулирования** открытым ключом **родительского сертификата**.

f) **Хост** ECI обрабатывает любые байты расширения в **списке аннулирования**, если он способен их обработать.

g) Хост **ECI** проверяет, не аннулируются ли **последующие** поля <entity type, entity id, version> в цепочке в соответствии со **списком аннулирования**, и устанавливает минимальную версию списка аннулирования, применимую к данному **сертификату**.

2) **Хост ECI** выполняет указанные ниже этапы предварительной проверки следующего **сертификата**.

a) **Хост ECI** проверяет версию **сертификата**. Если версия не соответствует функциональным возможностям хоста по обработке, загрузка цепочки прерывается.

b) **Хост ECI** проверяет поле типа идентификатора сертификата. Если поле не соответствует предполагаемым значениям, проверка прерывается.

c) **Хост** ECI проверяет соответствие длины **сертификата** определению его формата.

d) **Хост ECI** проверяет подпись **сертификата** открытым ключом **родительского сертификата**.

e) **Хост ECI** обрабатывает любые дополнительные поля и/или байты расширения **сертификата**, если он способен их обработать.

Цепочка **списка аннулирования**, извлеченная из дерева **списка аннулирования**, может использоваться для проверки аннулирования определенного элемента, который должен быть загружен **усовершенствованной системой безопасности**. Такой элемент можно идентифицировать с помощью последовательности идентификаторов **сертификатов**, используемых для его проверки при загрузке в **усовершенствованную систему безопасности**. Правила обработки цепочки **списка аннулирования** по умолчанию идентичны правилам обработки **цепочки сертификатов**.

3) **Подсистема CPS** загружает текущий **список аннулирования** и поля <entity type, entity id, version> следующего **сертификата** в CPS. Подсистема CPS выполняет следующую проверку.

a) Подсистема CPS проверяет поле format_version списка аннулирования на соответствие версии, которую он может интерпретировать, и поля rl_id.type и rl_id.rl_indicator на соответствие предполагаемым значениям.

b) Подсистема CPS выбирает корневой сертификат с root_version в качестве родительского объекта, если родительский объект является корневым сертификатом (root_version_indicator=1). В противном случае используется предварительно загруженный или предыдущий сертификат.

c) Подсистема CPS проверяет подпись списка аннулирования открытым ключом родительского сертификата.

d) Подсистема CPS проверяет, соответствует ли длина списка аннулирования значениям его полей.

e) Подсистема CPS проверяет, не является ли номер версии списка аннулирования недействительным.

f) Подсистема CPS проверяет, не аннулируются ли в цепочке (в соответствии со списком аннулирования) следующие поля <entity type, entity id, and version> и устанавливает минимальную версию списка аннулирования, сопутствующего данному **сертификату**.

4) Затем **хост ECI** загружает **сертификат** в соответствующей точке обработки CPS, которая выполняет следующие проверки.

a) Подсистема CPS проверяет поле format_version списка аннулирования на соответствие версии, которую он может интерпретировать, и поле id.type и id.entity_id на соответствие предполагаемым значениям.

b) Подсистема CPS проверяет соответствие длины сертификата значениям его полей.

c) Подсистема CPS выполняет проверку подписи относительно открытого ключа родительского сертификата.

5.5 Наборы деревьев аннулирования и файлы данных аннулирования

Для проверки конкретного **объекта** следует выбирать данные аннулирования, содержащие список аннулирования **родительского объекта целевого объекта**.

При распространении данных аннулирования те цепочки, которые аннулируют несколько целевых объектов, могут объединяться в дерево, что позволяет избежать дублирования корневых и **дочерних сертификатов** и связанных с ними **списков аннулирования** и выполнять упорядоченный поиск в устройствах CPE.

Кроме того, деревья аннулирования могут быть легко объединены в наборы для упрощения формирования и расформирования массива данных аннулирования. Однако наборы деревьев не должны перекрываться, за исключением общего списка аннулирования **родительского объекта**. Наборы деревьев могут содержать несколько списков аннулирования корневых сертификатов (в процессе выгрузки изменений текущего корневого сертификата **доверительного органа**).

Определение **цепочки сертификатов** и дерева **списка аннулирования** должно выполняться в соответствии с таблицей 5.5-1.

Таблица 5.5-1 – Определение набора деревьев списка аннулирования

Синтаксис	Количество битов	Мнемоника
ECI_RL_Tree_Set {		
tree_number	32	uimsbf
для (i=0; i<tree_number; i++) {		
ECI_RL_Tree **tree**		
}	8	uimsbf
}		

Семантика

tree_number: integer	Количество деревьев в наборе
tree: ECI_RL_Tree	Дерево (включая **корневой сертификат**) **сертификатов** и их **списков аннулирования**.

ПРИМЕЧАНИЕ. – Для минимизации трафика данных онлайн-серверы могут распространять деревья, нацеленные на единичный объект (эффективные цепочки). В сетях радиовещания деревья могут быть легко разделены и объединены в соответствии с количеством сегментов (см. пункт 7.7.2), используемых в карусели передачи.

Деревья или наборы деревьев аннулирования не обязательно должны быть полными, то есть содержать все объекты класса. **Система управления платформой** может составить необходимый набор деревьев аннулирования, который обеспечивает минимальный риск в установленном оборудовании **CPE** в сети **системы управления платформой**. Кроме того, **списки аннулирования** могут чередоваться по времени в сетях радиовещания, для того чтобы расширить охват аннулирования.

ECI требует, чтобы оборудование **CPE** постоянно хранило цепочки **доверительного органа ECI** для всех потенциально загружаемых элементов, чтобы ранее аннулированные объекты оставались аннулированными. Данное требование описывается в соответствующих разделах.

В целях удобства транспортировки наборы деревьев аннулирования **ECI** группируются в формате, указанном в таблице 5.5-2.

Таблица 5.5-2 – Файл данных аннулирования

Синтаксис	Количество битов	Мнемоника
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**		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – три 8-битовых представления символов ERD в формате ASCII. **Хост ECI** проверяет значение данного поля, чтобы удостовериться, что формат файла **ECI** соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; все остальные значения зарезервированы. **Хост ECI** игнорирует все образы, номер версии которых не распознается
father_type: integer	Тип общего **родительского объекта** данных **списка аннулирования**. 0x0 обозначает **корневой сертификат ECI** Значения 0x1–0x7 зарезервированы. Значения 0x8-0xF могут применяться для частных приложений
sub_type: integer	Если поле father_type равно 0x0, этот параметр определяет тип общего **списка аннулирования** в соответствии с таблицей 5.2-2 **корневого сертификата ECI** для данных, относящихся к аннулированию. Для прочих значений father_type этот параметр не определен
revocation_data: ECI_RL_Tree_Set	Набор деревьев списков аннулирования для аннулируемых элементов

5.6 Подписи больших элементов данных

Используя эффективную хеш-функцию для хеширования массива данных в сочетании с обычным режимом подписания, **ECI** вычисляет подписи для больших массивов элементов данных (например, образов программ). Подпись больших массивов элементов данных определяется в таблице 5.6-1.

Таблица 5.6-1 – Определение подписи больших массивов элементов данных

Синтаксис	Количество битов	Мнемоника
ECI_Data_Signature {		
sign_version	8	uimsbf
padding(4)	24	uimsbf
если (sign_version == 0x01){		
для (i=0; i<256; i++){		
signature_byte	8	uimsbf
}		
}		
}		

Семантика

sign_version: integer	Версия подписи. Значение 0 × 01 является текущей версией; все остальные значения версий зарезервированы. Если в оборудовании **CPE** эта версия не используется, данное поле (а также все последующие данные) игнорируется)
signature_byte: byte	Последовательность байтов, представляющих подпись большого массива элементов

Алгоритм подписания определяется в Приложении А.

5.7 Корневые сертификаты

5.7.1 Определение корневого сертификата

ECI использует последовательность ***версий*** **корневого сертификата.** В случае если любой из предыдущих **списков аннулирования** для любого **дочернего объекта** слишком велик или если секретный ключ, связанный с открытым ключом **сертификата**, не считается достаточно защищенным, **доверительный орган ECI** может использовать новую версию **корневого сертификата**.

Корневой сертификат использует поле идентификатора **сертификатов ECI**. Определение поля приведено в таблице 5.7-1. Поля type и identifier не используются; используется только поле version.

Таблица 5.7-1 – Определение поля Root_ID ECI

Синтаксис	Количество битов	Мнемоника
ECI_Root_Id {		
type /* см. таблицу 5.2-1***/**	4	uimsbf
id /* см. таблицу 5.2-2***/**	20	uimsbf
version	8	uimsbf
}		

Семантика

version: integer	Номер версии **сертификата**; нумерация начинается с 1. Номер увеличивается на единицу с каждым новым выпуском **корневого сертификата**. Значение 0x00 зарезервировано

5.7.2 Управление корневым сертификатом хоста ECI

Доверительный орган ECI может начать использование нового **корневого сертификата** с более высоким номером версии. После этого в определенный момент времени он может выпускать **список аннулирования** для нового **корневого сертификата**, который аннулирует предыдущие **корневые сертификаты**. Все **сертификаты**, подписанные с помощью такого корневого сертификата, признаются недействительными.

В качестве альтернативы **доверительный орган ECI** может принять решение о том, что **список аннулирования** для объектов определенного типа (например, **производителей**) слишком велик, а также о перевыпуске новых версий выпущенных ранее **сертификатов**, используя поле с более высоким значением **min_id_version** в **списке аннулирования** для данного типа **объекта**. Тем самым все выпущенные ранее **сертификаты** для типов **объекта** вплоть до **min_entity_version-1** фактически признаются недействительными. Как правило, это влечет необходимость выпуска значительного числа новых **сертификатов** с более высоким номером версии для объектов, все еще использующих более низкую версию **сертификата**, взамен аннулируемых **сертификатов**.

Ресурсы, которые **хост ECI** предоставляет для хранения **корневых сертификатов**, предлагаются в [b-ITU-T J Suppl. 7].

6 Загрузчик хоста ECI

6.1 Введение

В процессе загрузки **хоста ECI** выделяются следующие аспекты:

1) хранение изображения, проверка подлинности образа оборудованием **CPE** с использованием предоставленных **доверительным органом ECI** аутентификационных данных и последующая активация образа;
2) формат файла (файлов), содержащего образ и любую другую информацию, необходимую для загрузки образа в оборудование **CPE**;
3) транспортный протокол для передачи **образа хоста ECI** в оборудование **CPE**. Это относится к любой точке расположения необходимых образов, обнаруженной с помощью оборудования **CPE**. Данный протокол включает в себя любое хранение переданных образов и дополнительных данных по цепочке подтверждения **ECI** и данных, относящихся к подписи;
4) любое аннулирование **образов хоста ECI**, выполняемое конкретным **оператором**; формат данных для подобной информации определяется в разделе 6, а порядок применения – в разделе 8.

Логика проверки и аутентификация образа применяются на недавно загруженных **образах хоста ECI** и на данных аутентификации при каждой перезагрузке оборудования **CPE**, а также там, где это предусмотрено при нормальном функционировании оборудования **CPE**.

6.2 Хранение, проверка и активация

6.2.1 Принципы работы

Хост ECI обеспечивает возможность запуска **клиентов ECI** в приватной и защищенной среде в соответствии с требованиями **ECI** к устойчивости системы при введении в действие таких клиентов. Кроме того, **хост ECI** предотвращает взаимные помехи между **клиентами ECI**. С этой целью **доверительный орган ECI** может сертифицировать программное обеспечение для устройств **CPE**, а загрузчик оборудования **CPE** проверяет подлинность образов программного обеспечения, которые он загружает.

Во многих устройствах **CPE** используются многоступенчатые загрузчики. Интерфейс **ECI** предусматривает, что перед началом загрузки любых стандартных образов программного обеспечения основная микросхема устройства **CPE** загружает ряд образов инициализации для конкретной микросхемы. В соответствии с лицензионным соглашением поставщика микросхем и **доверительного органа ECI** такие образы могут быть сертифицированы в неявной форме. Кроме того, они могут быть частью процесса сертификации **производителя**, который описывается в настоящем разделе.

Если программное обеспечение одного из образов, управляемых **ECI**, в последующем указывает на наличие ошибки в системе безопасности, **доверительный орган ECI** и **производитель CPE** могут аннулировать образ и заменить его версией с исправленной ошибкой.

На рисунке 6.2.1-1 предполагается, что **Img1** представляет собой образ для определенной микросхемы, необходимый для подготовки этой микросхемы к началу загрузки стандартных образов приложений. Образ защищается подписью **CS1** для определенной микросхемы. Подпись проверяется **загрузчиком микросхем** с использованием проприетарного ключа поставщика микросхем.

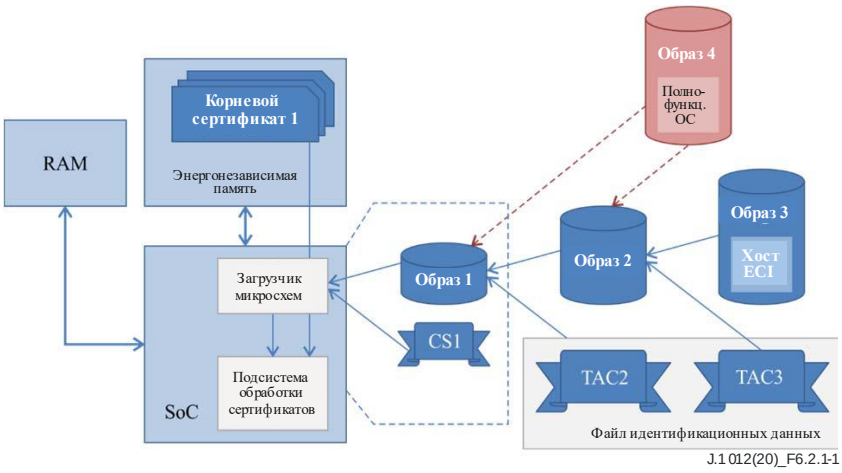


Рисунок 6.2.1-1 – Пример процесса загрузки хоста ECI

После запуска **Img1** микросхема переходит к загрузке других образов. Она загружает **Img2**, который может быть аутентифицирован с помощью **цепочки сертификатов** и подписи образа **TAC2**. Проверка образа выполняется с использованием **корневого сертификата доверительного органа**, **подсистемы обработки сертификатов** и **TAC2. Img2** приступает к загрузке **Img3**, который содержит программное обеспечение **хоста ECI**. **Img3** проверяется **подсистемой обработки сертификатов**, **корневыми сертификатами**, **цепочкой сертификатов** доверительного органа и подписью образа **TAC3**. Дополнительные образы, такие как **Img4**, содержащие, например, полнофункциональную операционную систему (rich OS) и не сертифицированные **доверительным органом ECI**, могут быть загружены, если среда загрузки может гарантировать, что это не создаст угрозу безопасности для **хоста ECI**.

Идентификационные данные, присвоенные образам доверительным органом, передаются в специальном файле.

Доверительный орган ECI удостоверяет целостность системы безопасности **хоста ECI**: его способность обеспечивать сохранность личных данных клиента и защиту **хоста ECI** от внешних угроз, а также предотвращать создание клиентами нежелательных взаимных помех друг другу. **Производители CPE** могут использовать дополнительные меры безопасности при загрузке образов с помощью собственных проприетарных механизмов шифрования и аутентификации образов.

Системы управления платформой могут проверять актуальность **образов хоста ECI** и принимать решение не проводить дешифрование услуг. С этой целью подсистема CPS извлекает минимальный номер версии **списка аннулирования**, который используется для проверки любого загружаемого элемента, тем самым позволяя **системе управления платформой** проверять приложение последнего **списка аннулирования**. Процедуры приемки для **хоста ECI**, связанные с определенной **системой управления платформой**, описываются в разделе 8.

Загрузчик хоста ECI хранит последние **образы** и последние идентификационные данные **хоста ECI** в энергонезависимой оперативной памяти. **Загрузчик хоста ECI** повторно проверяет каждый образ, который он загружает при перезагрузке **хоста ECI**. Данная процедура восстанавливает подлинность **хоста ECI** при каждой перезагрузке.

6.2.2 Определение идентификационных данных

6.2.2.1 Сертификаты, относящиеся к образам хоста ECI

Интерфейс **ECI** предоставляет два типа устройств **CPE ECI** в зависимости от **образа хоста ECI**.

1) **Устройства CPE** обобщенного типа, загружающие одинаковый набор **образов хоста ECI** на каждом экземпляре оборудования **CPE** одинакового типа и версии.
2) Устройства **CPE** персонализированного типа, загружающие (частично) различающийся набор образов на каждом экземпляре оборудования **CPE** одинакового типа и версии. Такого рода серия образов одного и того же типа, персонализированная для каждого устройства **CPE**, называется **серией образов (Image Series)**.

Цепочка сертификатов хоста ECI состоит из следующих **сертификатов** (каждый из которых сертифицируется предыдущим сертификатом).

1) Корневой сертификат:
   - представляет центральный корневой **объект доверительного органа ECI**. Открытый ключ этого **сертификата** используется для проверки.
2) Сертификат производителя:
   - представляет **объект доверительного центра ECI** для конкретного **производителя**. Открытый ключ этого **сертификата** используется для проверки.
3) Сертификат хоста:
   - представляет оборудование **CPE**, сертифицированное **доверительным органом ECI**, и версию программного обеспечения **хоста ECI**. Что касается обобщенных **хостов ECI**, открытый ключ этого **сертификата** используется для аутентификации всех **образов хоста ECI**. Что касается персонализированных **образов хоста ECI**, открытый ключ этого **сертификата** используется для проверки.

4) Сертификат серии образов хоста:

– данный **объект** предоставляет общее разрешение для серии образов, характерных для конкретной конфигурации оборудования CPE, но в остальном идентичных с точки зрения **доверительного органа ECI**. Что касается персонализированных **хостов ECI**, открытый ключ этого **сертификата** используется для аутентификации **образа хоста ECI**, предназначенного для конкретного устройства **CPE**, идентификатор которого соответствует идентификатору в **сертификате**.

ПРИМЕЧАНИЕ. – Каждый идентификатор объекта следует интерпретировать в контексте авторизующего объекта, то есть идентификаторы являются связанными.

На рисунке 6.2.2.1-1 представлены **образ хоста ECI** и структура сопутствующей сертификации, а в таблице 6.2.2.1-1 приведен обзор соответствующих параметров.

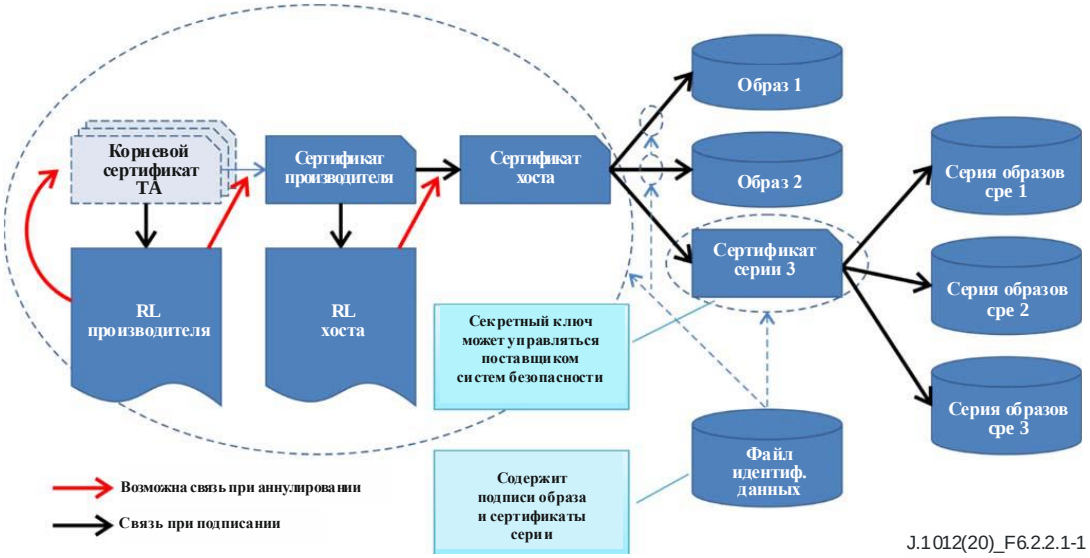


Рисунок 6.2.2.1-1 – Структура сертификации образа хоста ECI

Таблица 6.2.2.1-1 – Обзор параметров сертификатов, относящихся к хосту ECI

Тип	Объект	Значение поля ID сертификата	Порядок обработки хостом ECI
0x0	**Производитель**	manufacturer_id, version	Manufacturer_id проверяется на соответствие идентификатору производителя **CPE** в блоке AS
0x0	Хост	cpe_type, cpe_model, host_version	cpe_type и cope_model проверяются на соответствие типу и модели **CPE** блоком AS
0x8	**Серия образов** CPE	target_id	target_id проверяется на соответствие идентификатору **CPE**
0x8	Образ CPE	n.a.	
0x8	**Образ хоста ECI**	ECI_Host_Image_Id	Таким образом обозначается тип для действующей подписи образа

Определения для **сертификатов**, относящихся к **хосту ECI**, должны соответствовать общему определению **ECI_Certificate**, приведенному в пункте 5.2. Поля идентификатора **сертификатов** для управления **хостом ECI** определяются в таблице 6.2.2.1-2.

Таблица 6.2.2.1-2 – Определение поля идентификатора для сертификатов, относящихся к хостам

Синтаксис	Количество битов	Мнемоника
ECI_Manufacturer_Id {		
padding(4)		
type /* см. таблицу 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 /* см. таблицу 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 /* см. таблицу 5.2-2 */	4	uimsbf
image_series_model	8	uimsbf
image_series_model_extension	4	uimsbf
image_series_version	16	uimsbf
}		

Семантика

type	Значение в соответствии с таблицей 5.2-2
manufacturer_id: integer	Идентификатор, присвоенный **производителю доверительным органом ECI**
cpe_type: integer	Идентификатор, присвоенный модели **CPE** доверительным органом **ECI**. Значения 0x000 и 0x3F0..0x3FF зарезервированы. Устройства **CPE** одной модели должны обладать высокой степенью унифицированности и использовать одну и ту же технологию безопасности **ECI**
cpe_model: integer	Идентификатор, присвоенный версии конкретной модели, которая идентична другим во многих отношениях, но обладает рядом нетривиальных различий. Значение присваивается **доверительным органом ECI.** Значения 0x00 и 0xF0..0xFF резервируются
cpe_type_id: ECI_CPE_Type_id	Идентификатор типа оборудования **CPE** (версия + модель), уникальный в контексте manufacturer_id
cpe_host_version	Идентификатор, присвоенный набору образов, формирующих конфигурацию **хоста ECI** для оборудования **CPE**
image_series_model: integer	Идентификатор образов одного и того же типа для устройств **CPE**, поддерживающих **серию образов**; различие определяется значением **cpe_id**. Значения 0x000 и 0xF00..0xFFF резервируются
image_series_version: integer	Идентификатор с определенным шагом, присваиваемый **доверительным органом ECI** версии модели серии образов. Значения 0x0000 и 0xF000..0xFFFF резервируются

6.2.2.2 Подписи образа хоста ECI

Идентификатор **образа хоста ECI** должен быть равен идентификатору серии образов хоста и определяется в таблице 6.2.2.2-1.

Таблица 6.2.2.2-1 – Определения идентификаторов образа хоста и образа серии хоста

Синтаксис	Количество битов	Мнемоника
ECI_Host_Image_Id {		
padding(4)		
type /* см. таблицу 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
если (target type == 0x1){		
ECI_CPE_Id **cpe_id**	60	uimsbf
}		
}		

Семантика

type	Значение в соответствии с таблицей 5.2-2
image_model: integer	Идентификатор, присвоенный **образу хоста ECI** или серии образов, заменяющих друг друга. Значения 0x00 и 0xF0.0xFF резервируются
image_model_extension: integer	Расширение вышеуказанного поля. В стандартных приложениях значение данного поля должно быть задано равным 0x0
image_version: integer	Версия образа того же типа, присвоенная с определенным шагом. Значения 0x00 и 0xF0..0xFF резервируются
cpe_serial_number: integer	Серийный номер оборудования **CPE**, для которого предназначен образ. Значение cpe_serial_number должно быть уникальным в контексте <manufacturer_id, cpe_type_id>
cpe_type: integer	Поле cpe_type field, определяемое в структуре ECI_CPE_Type_Id
manufacturer_id: integer	Поле manufacturer_id field, определяемое в структуре ECI_Manufacturer_Id
target type: integer	Тип идентификации целевого объекта для образа серии. Значение 0x1 определяет данную структуру и указывает на то, что cpe_id используется как целевой объект; другие значения резервируются
cpe-id: ECI_CPE_Id	Идентификатор оборудования **CPE**, которое является целевым объектом образа серии (**хост ECI** или **клиент ECI**)

Подписи **образа хоста ECI** и подписи **серий образов хоста ECI**, применяемые для подписи действующих **образов хоста ECI**, используют структуру подписи больших массивов данных, определяемую в пункте 5.5.

6.2.2.3 Идентификационные данные хоста ECI

В таблице 6.2.2.3-1 определяется структура идентификационных данных **хоста ECI**, которая проверяет набор **образов хоста ECI**.

Таблица 6.2.2.3-1 – Определение структуры идентификационных данных хоста ECI

Синтаксис	Количество битов	Мнемоника
ECI_Host_Credentials{		
image_credential_version	8	uimsbf
если (image_credential_version == 0x01) {		
padding(4)	24	uimsbf
ECI_Certificate_Chain **image_chain**		
nr_images	8	uimsbf
padding(4)	24	uimsbf
для (i=0; i<images; i++){		
ECI_Host_Image_Id **image_id**	32	uimsbf
если (image_id.type == 0x8) {		
ECI_Certificate **series_cert**		
} или же если (image_id.type == 0x9){		
ECI_Data_signature **image_signature**		
}		
}		
Extension_Field **extension**		
}		
}		

Семантика

image_credential_version: byte	Версия формата идентификационных данных. Значение 0x01 является текущей заданной версией; все остальные значения резервируются. **Загрузчики хоста ECI** игнорируют любые идентификационные данные, кроме тех, значения которых они могут распознать
image_chain: ECI_Certificate_Chain	Двухуровневая **цепочка сертификатов**, от **корневого списка аннулирования производителя** до **сертификата хоста ECI**. Последний **сертификат** используется для проверки подписи образа для любого **сертификата серии образов**
nr_images: integer	Количество образов, для которых учитываются подписи
image_id	Идентификатор образа, для которого подпись следует в цикле Идентификаторы image_id, перечисленные в цикле, должны иметь разные значения поля **image_id.image_model**
series_cert: ECI_Certificate	Сертификат, используемый для проверки **серии образов**
image_signature: ECI_Data_Signature	Подпись образа (включая ID образа хоста)
extension: поле расширения	Поле обратно совместимого расширения

При проверке цепочки **image_chain** оборудование **CPE** подчиняется общим правилам обработки цепочек, определенным в пункте 5.4.

6.2.3 Процесс загрузки файла образа хоста ECI

Оборудование **CPE** хранит, проверяет и запускает выполнение набора файлов **образа хоста ECI**, необходимых для запуска **хоста ECI**. Как правило, фактическая активация **образа хоста ECI** происходит при загрузке оборудования **CPE**.

В целях загрузки, проверки и активации выбранного **образа хоста ECI** оборудование **CPE** использует устойчивую функцию обработки, называемую **загрузчиком хоста ECI**. Если, к примеру, загрузочный образ **CPE**, содержащий **загрузчик хоста ECI**, запускает выполнение второго образа, а второй образ загружает и запускает выполнение третьего образа, то функциональные возможности второго образа, заключающиеся в корректной загрузке третьей выполняемой проверки подписи образа, считаются набором функций **загрузчика хоста ECI** для данного оборудования **CPE**. **Образ хоста ECI** может проверяться и запускаться только функцией **загрузчика хоста ECI**. Для проверки идентификационных данных **загрузчик хоста ECI** использует **подсистему обработки сертификатов** (CPS).

Оборудование **CPE** хранит последний набор файлов **образа хоста ECI** и его идентификационные данные, загруженные в **энергонезависимую память**. При загрузке оборудования **CPE загрузчик хоста ECI** должен быть способен находить эту информацию и запускать загрузку образов способом, подходящим для конкретного типа оборудования **CPE**.

Для проверки каждого загруженного образа **загрузчик хоста ECI**, использующий подсистему обработки сертификатов (CPS), руководствуется стандартными правилами обработки цепочек согласно пункту 5.4. Обобщенные образы и **сертификаты серий образов** проверяются открытым ключом **сертификата** хоста. Открытый ключ **сертификата серии образов** применяется для проверки

серии образов. Оборудование **CPE** проверяет поле cpe_id образа на соответствие полю cpe_id оборудования **CPE**.

В случае повреждения образа (сбоя проверки подписи оборудованием CPS) **загрузчик хоста ECI** отклоняет образ. Это означает, что **хост ECI** не может быть реализован на оборудовании **CPE**. Оборудование **CPE** должно быть способно исправить ситуацию с помощью процедуры восстановления, заключающейся в повторной инициализации последнего **образа хоста ECI** и его идентификационных данных. Например, может быть применена повторная загрузка последнего набора файлов **образа хоста ECI** по радиовещательному каналу, с онлайн-сервера **образа хоста ECI** или с помощью ряда других средств.

Независимо от канала получения, **хост ECI** хранит последние полученные версии цепочки **сертификатов хоста ECI**. Фактически тем самым последний доступный **сертификат** хоста фиксируется в качестве основы для будущих проверок образов.

Последовательность загрузки **образов хоста ECI** не проверяется непосредственно в процессе проверки подписи: данная процедура выполняется загрузчиком для первого **образа хоста ECI**, а также для последующих активаций самими предшествующими **образами хоста ECI**.

6.3 Форматы файлов, связанных с хостом ECI

В настоящей Рекомендации не определяются наименования файлов или другие метаатрибуты для файлов **образа хоста ECI**. В нем рассматриваются данные **образа хоста ECI** в виде набора контейнеров данных (безымянных файлов, относящихся к **ECI**), идентифицируемых по собственному идентификатору образа хоста, и идентификационных данных **ECI** (**цепочек сертификатов** и подписей), необходимых для их аутентификации.

Файл **образа хоста ECI** должен быть последовательностью ECI_Host_Image_Header и контента образа. Он должен соответствовать определению, приведенному в таблице 6.3-1.

Таблица 6.3-1 – Определение файла образа хоста ECI

Синтаксис	Количество битов	Мнемоника
ECI_Host_Image_File {		
magic = 'EHI'	24	
image_header_version	8	uimsbf
если (image_header_version == 0x01) {		
ECI_Host_Image_Id **host_image_id**	32	uimsbf
ECI_Manufacturer_Id **manufacturer_id**	32	uimsbf
Extension_Field **extensions**		
для (i=0; i<n; i++) {		
host_image_byte	8	uimsbf
}		
}		
}		

Семантика

host_image_byte: byte	Действительный **образ хоста ECI**; собственный формат оборудования **CPE**
magic: byte[3]	Системный код используется для проверки формата следующих данных. Его значение – три 8-битовых представления символов 'EHI' в формате ASCII. Встроенное ПО оборудования **CPE** проверяет значение данного поля, чтобы удостовериться, что формат файла ECI соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; остальные значения версий зарезервированы
host_image_id: ECI_Host_Image_Id	Идентификатор **образа хоста ECI** для образа. Устройства **CPE** проверяют данное поле перед загрузкой (нового) **образа хоста ECI**
manufacturer_id: ECI_Manufacturer_Id	**ECI**_Manufacturer_ID **производителя** оборудования **CPE** для **образа хоста ECI**. Устройства **CPE** проверяют данное поле перед загрузкой (нового) **образа хоста ECI**. См. примечание
extensions: Extension_Field	См. пункт 5.1 настоящей Рекомендации: обратно совместимые расширения
host_image_byte: byte	Действующий **образ хоста ECI**
ПРИМЕЧАНИЕ. – Должно также соблюдаться соответствие OUI **производителя**, используемому в каруселях радиовещания для передачи надлежащего файла.	

Файлы **серии образов** имеют уникальную подпись, которая передается в самом файле образа. Поэтому конкретный формат файла должен соответствовать определению, указанному в таблице 6.3-2.

Таблица 6.3-2 – Определение файла серии образов хоста ECI

Синтаксис	Количество битов	Мнемоника
ECI_Host_Image_Series_File {		
magic = 'EHS'		
image_header_version	8	uimsbf
если (image_header_version == 0x01) {		
ECI_Data_Signature **image_signature**		
ECI_Image_Target_Id **target_id**	64	
Extension_Field **extensions**		
для (i=0; i<n; i++) {		
host_image_byte	8	uimsbf
}		
}		
}		

Семантика

host_image_byte: byte	Действительный **образ хоста ECI**; собственный формат оборудования **CPE**
magic: byte[10]	Системный код, используемый для проверки формата следующих данных. Его значение составляет три 8-битовых представления символов EHS в формате ASCII. Встроенное ПО оборудования **CPE** проверяет значение данного поля, чтобы удостовериться, что формат файла ECI соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; все другие значения резервируются
image_signature: ECI_Data_Signature	Подпись для всех данных, содержащихся в файле образа
target_id: ECI_Series_Image_Target_Id	Идентификатор целевого объекта для образа. Значение для target_id.target_type равно 0x01, все другие значения резервируются
extensions: Extension_Field	См. пункт 5.1: обратно совместимые расширения
host_image_byte: byte	Последовательность байтов, формирующих **образ** хоста

Идентификационные данные **образа хоста ECI** соответствуют указанному в таблице 6.3-3 определению, которое по сути является **цепочкой сертификатов** с набором подписей образа или **сертификатов серии образов**.

Таблица 6.3-3 – Определение файла идентификационных данных образа хоста ECI

Синтаксис	Количество битов	Мнемоника
ECI_Host_Image_Credential_File{		
magic = 'EHC'	24	uimsbf
version	8	uimsbf
если (version == 0x01) {		
ECI_Host_Credentials **credentials**		
}		
}		

Семантика

magic	Системный код, используемый для проверки формата следующих данных. Его значение составляет три 8-битовых представления символов EHC в формате ASCII. Встроенное ПО оборудования **CPE** проверяет значение данного поля, чтобы удостовериться, что формат файла ECI соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
version	Версия формата файла. Значение 0x01 является текущей заданной версией; все другие значения резервируются
credentials: ECI_Host_Credentials	Идентификационные данные для одного **образа** или группы образов **хоста ECI**

Host_image_id используется для идентификации подписей **доверительного органа ECI** касательно набора файлов **образа хоста ECI**, включающих полную загрузку в структуре идентификационных данных **ECI**.

Совместимые с **ECI** устройства **CPE** позволяют загружать другие проприетарные программные модули **CPE**, используя тот же самый транспортный протокол, что используется для файлов **образа хоста ECI**. Для таких образов не требуется специальный формат.

По каналам радиовещания удобно распространять данные аннулирования большого количества **хостов ECI** в виде одного большого файла. **Хосты ECI**, получающие такие данные, могут использовать этот метод для проверки собственного **сертификата хоста ECI**.

Файл данных аннулирования **хоста ECI** использует формат ECI_Revocation_Data_File, определяемый в таблице 5.5-2. Файл данных аннулирования **хоста ECI** использует значение father_type, равное 0x0 (**корневой сертификат**) и значение sub_type, равное типу списка аннулирования **производителя**. Файл Revocation_data соответствует следующему ограничению: список аннулирования конечных узлов в деревьях является списком аннулирования **хоста**.

6.4 Транспортные протоколы образа хоста ECI

6.4.1 Введение

В настоящей Рекомендации выделяются три способа рассылки **образа** хоста:

В настоящей Рекомендации выделяются три способа рассылки **образа** хоста.

1) **Широковещательная передача** – интерфейс **ECI** определяет протоколы, позволяющие операторам **платформ** рассылать оповещения и доставлять новые файлы **образа хоста ECI** от **производителя** оборудования **CPE** действующим устройствам **CPE** посредством DVB-SSU.
2) **Доставка в режиме онлайн** – интерфейс **ECI** разрешает устройствам **CPE**, подключенным к интернету, загружать файлы **образа хоста ECI** посредством любого проприетарного протокола, предполагающего использование HTTP 1.1, а также посредством определяемого **ECI** интерфейса на веб-сервер, предоставленный оператором.
3) **Другие способы** – кроме того, производители **оборудования CPE** и/или **операторы** могут использовать другие методы рассылки файлов **образа хоста ECI**, в том числе офлайн-методы, например, через карты памяти (флешки) USB. Эти методы рассылки образов выходят за рамки настоящей Рекомендации. Тем не менее образы, загруженные посредством такого протокола, должны соответствовать формату файла и проверке образа согласно пунктам 6.3 и 6.2.

Устройства **CPE**, предназначенные для получения **услуг** сетей цифрового радиовещания, реализуют транспортный радиовещательный протокол передачи **образа хоста ECI**, определяемый в пункте 6.4.2.

Устройства **CPE**, подключенные по IP-протоколу, реализуют транспортный протокол Интернет передачи в режиме онлайн **образа хоста ECI**, определяемый в пункте 6.4.3, а также протокол, определяемый в пункте 7.7.3.3.

Оборудование **CPE** может использовать любой дополнительный транспортный протокол **образа хоста ECI**, в том числе радиовещательную передачу **хоста ECI** и транспортные офлайн-протоколы (например, USB-карты памяти). Принимая во внимание реальные сценарии использования, при которых некоторые из сетевых подключений недоступны, **производитель** оборудования **CPE** в любом случае предоставляет практические средства обновления **хоста ECI** на местах посредством комбинации вышеуказанных транспортных протоколов.

6.4.2 Радиовещательный транспортный протокол хоста ECI

6.4.2.1 Общие сведения и профилирование

Радиовещательный транспортный протокол **хоста ECI** позволяет передавать новые файлы **образа хоста ECI** и связанные с ними данные от **производителя CPE** устройствам **CPE** через инфраструктуру головной радиовещательной станции **оператора**. Кроме того, данный протокол допускает транспортировку файлов, не содержащих **образ хоста ECI** (для функций, не критичных к обеспечению безопасности). **Оператор** может играть активную роль, контролируя версии программного обеспечения на оборудовании **CPE**. Устанавливая стандарты для точек технической функциональной совместимости, настоящий протокол упрощает взаимодействие между **производителем** оборудования CPE и **оператором**:

- произвольная стандартная эстафетная (то есть в режиме хэндовер) передача данных загрузки от **производителя CPE оператору**;

ПРИМЕЧАНИЕ. – Подробная техническая информация по такому режиму хэндовера выходит за рамки спецификаций **ECI**.

- стандартный радиовещательный транспортный протокол (обеспечивающий единые условия перегона программного потока на головной широковещательной станции **оператора**);
- стандартное обнаружение, реализация транспортного протокола и выбор параметров рабочего транспортного протокола в приемниках.

Введенные в действе радиовещательный транспортный поток (TS) **хоста ECI** и оборудование **CPE** соответствуют DVB SSU [ETSI TS 102 006] и, как следствие, соответствуют разделу, в котором даны определения карусели передачи данных DVB [ETSI EN 301 192] и указания по реализации [ETSI TR 101 202], а также определению карусели передачи данных MPEG [ISO/IEC 13818-6].

Операторы и устройства **CPE** должны поддерживать простой профиль DVB-SSU, а в некоторых случаях – профиль DVB-SSU UNT.

Операторы могут поддерживать несколько синхронизированных каруселей.

Устройства **CPE** сканируют все карусели, объявленные соответствующим образом в SI, UNT (если применимо) и в PMT для соответствующих элементов загрузки.

Общая схема радиовещательной передачи для загрузки образов изображена на рисунке 6.4.2.1-1.

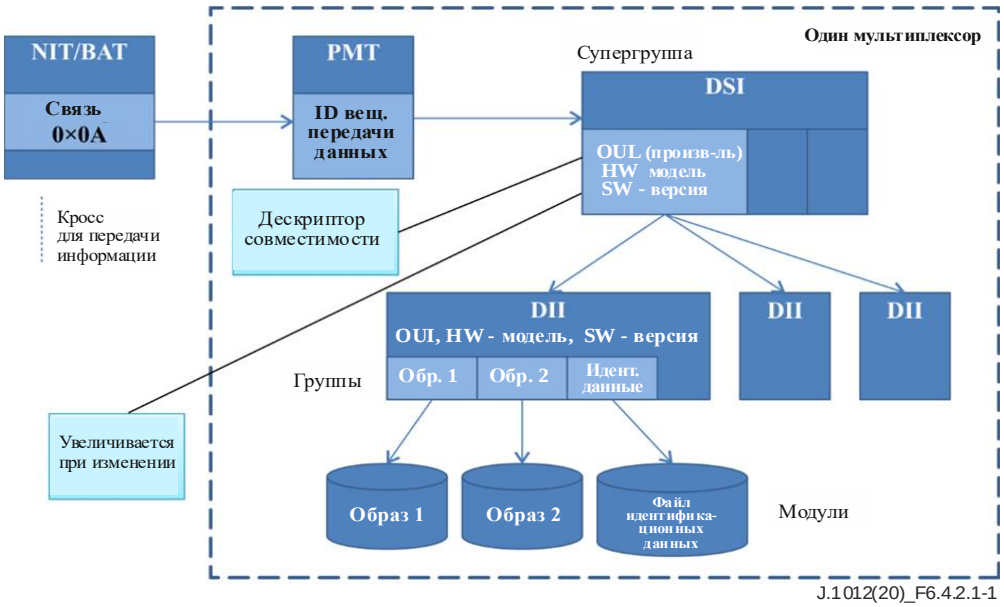


Рисунок 6.4.2.1-1 – Общая схема передачи сигналов образа хоста и структуры карусели (без варианта UNT)

6.4.2.2 Передача в режиме хэндовера от производителя CPE оператору

Любая будущая экосистема на базе **ECI** должна будет определять руководящие принципы для **операторов** и **производителей CPE**, чтобы обеспечить единый способ обмена информацией с файлами образов (как связанных, так и не связанных с **образами хоста ECI**), идентификационными данными образа **ECI** и метаданными, относящимися к загрузке от (нескольких) **производителей оборудования CPE** (нескольким) **операторам**.

6.4.2.3 Передача сигналов DVB SI

6.4.2.3.1 Передача сигналов о местоположении загрузки

Операторы должны поддерживать дескриптор связи DVB-SSU (тип связи 0 × 09) как минимум с общим DVB OUI (то есть связь со всеми каруселями без привязки к конкретному **производителю**) во всех NIT (наземных или кабельных) или в таблицах BAT (спутниковых).

Простой профиль устройств **CPE** должен поддерживать дескриптор связи DVB-SSU (тип связи 0 × 09).

Операторы, поддерживающие профиль DVB-SSU UNT, должны поддерживать дескриптор связи сканирования SSU (тип связи 0 × A) во всех NIT (наземных или кабельных) или в таблицах BAT (спутниковых).

UNT-профиль устройств **CPE** поддерживает дескриптор связи сканирования DVB-SSU (тип связи 0 × 09).

6.4.2.3.2 Срочные обновления

Для обозначения необходимости срочной замены **образа хоста ECI** один или несколько дескрипторов ECI_host_emergency_download могут быть помещены в таблицы NIT, BAT или в одну из записей SDT для услуги, доступ к которой может обеспечить доступ выделенный **хост ECI**. **Хост ECI** должен быть способен извлечь этот дескриптор из любой таблицы, в которой он появляется, в любом из настроенных в настоящий момент мультиплексов и выполнить соответствующую обработку, а также использовать любой резервный тюнер для доступа к соответствующим мультиплексам, чтобы собрать данные от этого дескриптора в худшем случае в течение 30 минут в период включенного питания. Ненастроенные мультиплексы рекомендуется проверять чаще (3-минутный интервал).

Дескриптор ECI_host_emergency_download_descriptor разрешает назначать целевыми объектами конкретные операционные платформы и конкретные **системы управления платформами**, а также образы клиентов в целях минимизации количества **пользователей**, испытывающих нарушения нормальной работы, вызванные срочными обновлениями.

Когда **хост ECI** обнаруживает новый дескриптор ECI_host_emergency_download, он должен соответствовать конфигурации **хоста ECI** и **клиента ECI** в отношении запланированных данных, заданных в дескрипторе. Если обнаружено совпадение с целевыми показателями и версия установленного на этот момент образа хоста требует обновления, **хост ECI** выполняет такое обновление в соответствии с emergency_indicator. Это вызовет нарушение в текущей работе **пользователей** на оборудовании **CPE**.

Операционный дескриптор **ECI** является частным дескриптором DVB. В таблице ему всегда должен предшествовать дескриптор DVB private_data_specifier_descriptor (см. [ETSI EN 300 468] и [ETSI TS 101 211]) с использованием поля **ECI** private_data_specifier_field. Синтаксис дескриптора определяется в таблице 6.4.2.3.2-1.

Таблица 6.4.2.3.2-1 – Дескриптор ECI_host_emergency_download_descriptor

Синтаксис	Количество битов	Мнемоника
ECI_host_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* основной цикл */		
main_loop_nr	8	uimsbf
для (i=0; i<main_loop_nr; i++){		
/* цикл клиента */		
client_nr		
для (j=0; j<client_nr; j++){		
platform_operation_tag	8	uimsbf
зарезервировано	3	
client_flag	1	
client_tag	4	uimsbf
}		
/* цикл образа хоста */		
host_nr	8	uimsbf
для (j=0; j<host_nr; j++){		
зарезервировано	4	
emergency_indicator	4	uimsbf
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
min_host_version	8	uimsbf
}		
}		
/* частные данные до конца дескриптора*/		
для (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Семантика

descriptor_tag	Значение маркера частных данных **ECI** для descriptor_tag: см. [b-ITU-T J Suppl. 7]
descriptor_length	См. [ETSI EN 300 468]
main_loop_nr	Количество записей в основном цикле. Отдельные записи основного цикла должны оцениваться раздельно **хостом ECI**, то есть обладать семантикой OR. Различные элементы одной записи цикла должны обладать семантикой AND
client_nr	Количество записей в цикле назначения клиента; значение 0x00 должно обозначать соответствие для любого клиента. Отдельные записи цикла должны обладать семантикой OR, и все клиенты, которые отвечают условию соответствия, должны учитываться при срочном обновлении. Поля одной записи цикла должны обладать семантикой AND
platform_operation_tag	Значение маркера для **системы управления платформой ECI**, отмеченное в ECI_platform_operation_descriptor в таблице NIT/BAT. **Хост ECI** должен предусматривать срочное обновление, если platform_operation соответствует platform_operation одного из установленных клиентов
client_flag	Отправляет сигнал о том, подходит ли поле client_tag field для сопоставления. Значение 0b0 означает неподходящий вариант (то есть любое значение client_id будет соответствовать), значение 0b1 означает что значение client_tag является подходящим
host_tag	Значение маркера, идентифицирующее **хост ECI** согласно списку в ECI_platform_operation_descriptor в таблице NIT/BAT, которое соответствует полю platform_operation_tag в той же записи цикла клиента. **Хост ECI** должен предусматривать срочное обновление, если рассматриваемые vendor_id и client_id соответствуют одному из установленных клиентов в **хосте ECI** для **системы управления платформой**
host_nr	Количество записей в цикле хоста. Минимальное значение должно быть равно 1. Записи цикла должны обладать семантикой OR; то есть если какая-либо спецификация хоста соответствует целевому условию, основной цикл находится в состоянии соответствия
emergency_indicator	**Хост ECI** использует значение данного поля в целях выбора соответствующего режима для начала загрузки и последующего обновления хоста, как указано в таблице 6.4.2.3.2-2
manufacturer_id	Manufacturer_id хоста, для которого предназначено срочное обновление. **Хост ECI** должен предусматривать срочное обновление, если значение данного поля соответствует manufacturer_id **хоста ECI**

cpe_type_id	Значение, определяемое идентификатором ECI_CPE_Type_ID в таблице 6.2.2.1-2. **Хост ECI** должен предусматривать срочное обновление, если cpe_type_id хоста соответствует значению данного поля. Если значение cpe_type_id.cpe_type равно 0x000, это означает соответствие cpe_types любого **хоста ECI** (а cpe_model и host-version следует игнорировать). Если значение cpe_type_id.cpe_model равно 0x00, это означает соответствие cpe_model любого хоста **ECI** (и версию хоста следует игнорировать)
min_host_version	**Хост ECI** должен предусматривать срочное обновление только в том случае, если его версия хоста меньше или равна значению данного поля. ПРИМЕЧАНИЕ. – Значение поля, равное 0xFF, предполагает соответствие всех версий хоста
private_data_byte	Частные данные: контент может определяться **оператором**, управляющим вещательной передачей этого дескриптора

В таблице 6.4.2.3.2-1 определяется ряд условий в основном цикле (обладающем семантикой AND), которые должны соблюдаться, чтобы **хост ECI** смог выполнить срочное обновление. Если все эти условия выполнены, **хост ECI** осуществляет срочную загрузку и установку нового образа хоста в соответствии с полем emergency_indicator для этого **хоста ECI**. Значения полей индикатора определяются в таблице 6.4.2.3.2-2.

Таблица 6.4.2.3.2-2 – Значения поля ECI_host_emergency_download_descriptor emergency_indicator

Название	Значение	Описание
Системная срочность	0x01	**Хост ECI** загружает новый образ хоста и устанавливает его как можно быстрее. При этом в случае необходимости допускается прерывание текущего сеанса работы **пользователя**. См. примечание
Обычная **срочность**	0x03	**Хост ECI** загружает новый образ хоста и устанавливает его при первой возможности, не нарушая текущий сеанс работы **пользователя. Хост ECI** загружает новый образ хоста самое позднее в ходе следующего включения оборудования. ПРИМЕЧАНИЕ. – **Операторы платформ** могут применять данные сценарии, например, в том случае, если действующий **хост ECI** испытывает серьезный недостаток услуг дешифрования, однако может функционировать надлежащим образом в обычных сценариях использования
RFU	Прочее	Зарезервировано для использования в будущем
ПРИМЕЧАНИЕ. – **Операторы платформ** могут применять данные сценарии, например, в том случае, если функционирование действующего **хоста ECI** совместно с назначенной платформой/клиентом существенно затруднено.		

6.4.2.4 Передача сигналов PSI

Операторы должны поддерживать дескриптор data_broadcast_id_descriptor в PMT [ETSI EN 300 468] для каждой передаваемой карусели; при этом от них не требовалось поддерживать любую передачу сигналов OUI в байтах селектора данного дескриптора.

Устройства **CPE** с упрощенным профилем SSU используют data_broadcast_id_descriptor в целях определения местоположения PID-потока, содержащего карусель DVB-SSU.

6.4.2.5 Вариант с использованием UNT

В данном разделе рассматриваются только устройства **CPE** и **операторы**, поддерживающие профиль UNT.

В PMT используется дескриптор data_broadcast_id_descriptor, содержащий структуру system_software_update_info с update_type 0×2 и поле OUI с заданным значением DVB OUI 0×00015A.

Операторы вносят записи в соответствующие таблицы SSU для каждого типа поддерживаемого оборудования **CPE**.

Хосты ECI должны быть способны интерпретировать следующие дескрипторы UNT (см. [ETSI TS 102 006]):

- SSU_location_descriptor (если карусель для данного типа оборудования **CPE** передается в текущий момент);
- Scheduling_descriptor (если передача карусели для данного типа оборудования **CPE** планируется в обозримом будущем);
- Message_descriptor.

Устройства **CPE** должны быть способны на постоянной основе выполнять полную загрузку принимаемой карусели, практически не содержащей ошибок, которую устанавливают и деинсталлируют в номинально обозначенное время и которая совершает два полных цикла (повторение всех сообщений в карусели) при условии отсутствия **пользовательских** действий, препятствующих загрузке.

6.4.2.6 Структура карусели

Карусели DVB SSU **ECI** (подробнее см. в [ETSI TS 102 006]) используют двухуровневые карусели передачи данных.

Карусель DVB SSU **ECI** использует сообщение DSI со следующими ограничениями:

- Для выполнения загрузки необходим полный список всех доступных групп.
- Каждая группа должна соответствовать одному полю **cpe_type + cpe_model** одного **производителя** и содержать все ресурсы **хоста ECI** данного типа оборудования **CPE**. Это предполагает, что максимально доступное число модулей равно 255 (файлы образов плюс один файл для идентификационных данных).

ПРИМЕЧАНИЕ 1. – Ввиду ограничений в значениях поля **ECI_host_id.model_id** предельное количество модулей составляет 239.

- Дескриптор CompatibilityDescriptor в поле GroupCompatibility, относящемся к структуре GroupInfoIndication (подробнее см. [ETSI TS 102 006]), использует следующее допущение:
  - цикл должен содержать системный аппаратный дескриптор:
    - идентификатор OUI должен соответствовать **производителю** оборудования **CPE**;
    - поля model и version, связанные с системным аппаратным дескриптором, должны соответствовать **cpe_type** и **cpe_model** оборудования **CPE** и быть равными полям **id.cpe_type** и **id.cpe_model сертификата хоста ECI** в файле идентификационных данных группы;
  - цикл должен включать в себя системный программный дескриптор; поле model должно быть установлено на 0, поле version должно отражать версию общего программного обеспечения **хоста ECI** в группе (то есть как связанных, так и не связанных с **образами хостов ECI**).

В целях соответствия собственным моделям и версиям устройства **CPE** используют поля model и version в дескрипторе compatibilityDescriptor. В целях проверки наличия обновления в группе устройства **CPE** используют поле version программного обеспечения. При наличии новых версий запускается загрузка новых образов.

Карусель DVB SSU **ECI** использует поля сообщений DII со следующими ограничениями:

- значение blockSize должно быть установлено на 2 кбайта (2048 байтов) как минимум;
- в поле tDownloadScenario должно быть введено отличное от нуля значение, которое отражает загрузку всех модулей с временем повтора, превышающим как минимум в 4 раза время повтора самого медленного сообщения (период оборота карусели);
- поле moduleId bits 7.0 должно быть равным полю **id.image_model** файла образа;
- поле moduleVersion должно быть равным полю **ECI id.image_version** файла образа.

Устройства **CPE** могут использовать поле tDownloadScenario для прекращения загрузок (прерванных, например, из-за частых ошибок в пакетах) и отправки **пользователю** сообщения об ошибках.

В группу типа оборудования **CPE** должны быть включены следующие модули:

- файлы образов для типа оборудования **CPE** (могут представлять собой частичный набор образов);
- файл идентификационных данных **образа хоста ECI**, содержащий последние данные для всех образов **хоста ECI**:

– данный модуль содержит DII moduleId bits 7..0 со значением, равным 0xFF; и

– moduleVersion увеличивается при каждом изменении.

ПРИМЕЧАНИЕ 2. – **Операторы** могут совместно использовать общие файлы между загрузками для разных типов оборудования **CPE** путем совместного использования поля DownloadDataBlocks между сообщениями DII. Однако это предполагает необходимость согласованного управления идентификаторами **образов хоста ECI** для разных типов оборудования **CPE**.

6.4.2.7 Процедура загрузки хоста ECI

Загрузчик **образов хоста ECI** при наличии доступа к сетевым ресурсам предпринимает попытки проверки всех возможных каруселей каждые 30 минут во включенном состоянии и не реже чем каждые 6 часов в режиме ожидания, не нарушая работу **пользователя**, например после переключения оборудования **CPE** в режим ожидания и в периоды просмотра не в пиковые часы.

Если поставщик сетевых услуг предоставляет доступ к таблицам UNT, которые могут содержать данные потенциальных загрузок для определенного типа оборудования **CPE**, то соответствующее оборудование **CPE** регулярно проверяет таблицу UNT на наличие расписания возможного обновления. Оборудование **CPE** предпринимает попытки проверки с той же периодичностью, что и для каруселей образов **хоста ECI**.

Рекомендуется предупреждать **пользователя**, если оборудование **CPE**, работающее только в вещательном режиме, не может выполнить вышеупомянутые проверки в течение более чем 2 недель.

Как только обнаружены новые данные для загрузки, что означает подтверждение со стороны оборудования **CPE** и **пользователя**, оборудование **CPE** предпринимает попытку выполнить загрузку и установить новый образ (возможно, перезаписав предыдущую версию). Если в процессе загрузки постоянно возникают сбои, об этом следует надлежащим образом уведомить **пользователя**. После неудачной загрузки образов **хосты ECI** должны быть способны восстанавливаться до функционального состояния, например, путем восстановления предыдущего образа хоста или попытки повторной загрузки нового образа хоста.

Следует отметить, что постоянная ошибка при загрузке новых **образов хоста ECI** или идентификационных данных может привести к отказу в обслуживании **оператором**.

6.4.2.8 Расписание каруселей, устанавливаемое оператором

Для выполнения загрузки в подходящее время **операторы** должны обеспечивать достаточную полосу пропускания для каруселей передачи данных образов **CPE**.

6.4.2.9 Аспекты пользовательского интерфейса

Оборудование **CPE**, способное выполнять загрузки **образов хоста ECI** через сети радиовещания, должно:

- иметь режим сканирования загрузки, который позволит автоматизировать регулярную проверку на наличие новых образов хоста или идентификационных данных, например находясь в режиме ожидания; рекомендуется, чтобы **производитель** устанавливал данный режим по умолчанию для проверки загрузки; а также
- иметь в меню **CPE** функцию, которая позволит подвергать автоматической обработке любое выданное пользователем разрешение на прием новых файлов **образа хоста ECI** или идентификационных данных; рекомендуется, чтобы **производитель** устанавливал данный режим по умолчанию для разрешения производить загрузку.

Устройства **CPE** должны предоставлять как минимум один альтернативный механизм для загрузки новых файлов **образа хоста ECI** в целях предотвращения функционирования устройств **CPE** в сетях радиовещания, которые не предоставляют новые файлы **образа хоста ECI** для собственного оборудования типа **CPE** в случае отказа в обслуживании.

6.4.3 Транспортный протокол Интернет хоста ECI

6.4.3.1 IP-протокол

Интерфейс **ECI** не определяет конкретный протокол для оборудования **CPE** в целях проверки новых файлов **образа хоста ECI** в результате услуги, предоставляемой **производителем**. Тем не менее рекомендуется использовать протокол HTTP1.1 [IETF RFC 7231] в качестве протокола передачи файлов. Как указано в пункте 7.7.3.3, этот протокол может быть использован для оказания стандартной услуги по загрузке файлов **образа хоста ECI** от сервера **системы управления платформой**.

Как правило, сервер загрузки **образа хоста ECI** предоставляется **производителем CPE**. При наличии особых договоренностей между **производителем CPE** и **оператором** (или третьими сторонами, действующими от их имени) протокол и сервер могут быть также предоставлены **оператором** или третьей стороной.

6.4.3.2 Работа загрузчика в режиме онлайн

З**агрузчик образа хоста ECI** в режиме онлайн предпринимает попытки проверять свой онлайн-сервер каждые 30 минут, не создавая помех действиям **пользователя**. Рекомендуется предупреждать **пользователя**, если оборудование **CPE**, работающее только в режиме онлайн, не допускает выполнения вышеупомянутых проверок в течение более длительного периода.

Как только обнаружены новые данные для загрузки, оборудование **CPE** предпринимает попытку выполнить загрузку и установить новый образ (при необходимости перезаписывая предыдущие версии образа). Если в процессе загрузки постоянно возникают сбои, об этом следует надлежащим образом уведомить **пользователя**.

Следует отметить, что ошибка при загрузке новых **образов хоста ECI** или идентификационных данных может привести к отказу в обслуживании **оператором**.

Загрузчик оборудования **CPE** в режиме онлайн рассылает набор (новых) образов и идентификационных данных образа, как определяется в пункте 6.3, для проверки, хранения и активации.

Загрузчик образа хоста **ECI** в режиме онлайн предоставляет функции срочной загрузки, обеспечивающие тот же результат, что определяется в пункте 6.4.2.3.2 для радиовещания.

6.4.4 Альтернативные транспортные протоколы

Хост ECI вправе использовать любые альтернативные (проприетарные) протоколы рассылки.

Загрузчик **CPE** обрабатывает набор (новых) образов и идентификационных данных образа, как определяется в пункте 6.3, для проверки, хранения и активации.

7 Загрузчик клиента ECI

7.1 Введение

Хост ECI может загружать, хранить и активировать **образы клиентов ECI** и сопутствующие данные. Процесс загрузки **клиента ECI** может быть разделен на несколько этапов.

1) Обнаружение защиты услуги/пакета услуг на базе **ECI** и/или другие способы определения потребности в **клиенте ECI**. Это является частью обычного применения оборудования **CPE** в целях навигации.
2) Определение сетевого местоположения (при помощи радиовещания или в режиме онлайн) ресурсов, необходимых для установки клиента **ECI** на **хосте ECI**.
3) Загрузка и хранение (в энергонезависимой памяти) информации **системы управления платформой**, необходимой для установки **клиента ECI**, а также проверка идентификационных данных.
4) Регистрация **хоста ECI** в системе безопасности **системы управления платформой** и получение (в случае необходимости) конкретных данных инициализации **CPE** для дешифрования **клиента ECI**.

5) Загрузка из сети и хранение (в энергонезависимой памяти) **образа клиента ECI** и соответствующих идентификационных данных **клиента ECI**, а также проверка идентификационных данных и образа, хранящихся в энергонезависимой памяти для использования в будущем.

6) Инициализация **клиента ECI** с использованием **образа клиента ECI,** сертификат **системы управления платформой**, распределение контейнера **ECI** и требуемых ресурсов системы AS, а также запуск выполнения клиента **ECI**.

Все процессы могут выполняться с использованием данных, полученных из радиовещательного потока или через интернет, за исключением регистрации оборудования **CPE** у **оператора**, которая требует проведения операций вручную в том случае, если доступна только радиовещательная связь.

Операторы могут обновлять ресурсы **клиента ECI** в любое время, публикуя информацию в радиовещательных сетях или через интернет. **Хост ECI** регулярно проверяет подобные обновления.

Интерфейс **ECI** требует наличия вспомогательных данных для различных функций оборудования **CPE**, например данных аннулирования или обновленных **цепочек сертификатов**, необходимых **клиенту ECI** и/или **хосту ECI** для поддержки **клиента ECI**. В сетях радиовещания транспортный протокол позволяет проводить выборочную загрузку данных, необходимых для оборудования **CPE**, на основе индекса (хеша) идентификации данных. Группирование данных по хеш-индексу называется "сегментное группирование" (bucketizing). В сетях онлайн выборочная загрузка основана на проведении идентификации требуемых данных в качестве параметра для API-интерфейса веб-услуг.

Следующие элементы данных могут загружаться **хостом ECI**:

- **образы клиента ECI** (в сгруппированном формате в сетях радиовещания);
- данные аннулирования **клиента ECI** (в сгруппированном формате в сетях радиовещания);
- цепочка клиентов системы управления платформой;
- данные аннулирования **системы управления платформой** (в сгруппированном формате в сетях радиовещания);
- **данные** аннулирования **образов хоста ECI** (в сгруппированном формате в сетях радиовещания). Данные инициализации клиента настройки **AS ECI** для дешифрования зашифрованных образов клиентов (в сгруппированном формате в сетях радиовещания).

7.2 Обнаружение клиентов ECI

7.2.1 Введение

Как правило, производители не устанавливают **клиентов ECI** на оборудование **CPE**, поддерживающее **ECI** (например, устройства iDTV), поскольку такие устройства имеются повсеместно в свободной продаже. В следующем разделе описываются доступные механизмы, позволяющие **ECI-**совместимым устройствам **CPE** обнаруживать **клиентов ECI,** которые могут потребоваться для дескремблирования услуг, поставляемых сетью, к которой подключено устройство.

Для проведения процедуры обнаружения можно выделить сети двух типов:

1) сети на основе транспортных потоков (сети радиовещания и типового IP-телевидения);

2) сети на основе IP-протокола.

Интерфейс **ECI** поддерживает два режима обнаружения поставщиков и клиентов в сетях на основе транспортных потоков:

1) ручная установка, в том числе параметры настройки основной (радиовещательной) сети;

2) автоматическое обнаружение (по выбору **пользователя**) – предполагается автономная установка устройства **CPE** для работы в сети.

Оба протокола – ручной установки и автоматического обнаружения в сетях на основе транспортных потоков – используют общую сигнализацию.

В сетях на основе IP-протокола интерфейс **ECI** поддерживает: ручной ввод base-URL.

7.2.2 Сети на основе транспортных потоков

7.2.2.1 Общая сигнализация

В целях сокращения количества параметров, которые **пользователь** должен вводить вручную, интерфейс **ECI** в режиме онлайн предусматривает сигнализацию ключевых параметров **ECI** для установки клиента:

- один или несколько дескрипторов ECI_ platform_operation_descriptors в таблице NIT, содержащей доступных клиентов (по идентификатору) для каждой **системы Platform_Operation**. Дескриптор включает в себя наименование оператора платформы и короткий идентификатор short-id (разрешающий компактное представление данных в строке ручной установки);
- поставщик платформы может задать базовый URL для веб-интерфейса API в дескрипторе ECI_base_URL_descriptor.

7.2.2.2 Дескриптор ECI_ platform_operation _descriptor

Дескриптор ECI_ platform_operation_descriptor предоставляет ключевую информацию о **системе Platform_Operation**, предлагающей услуги доступа в сетях на основе транспортных потоков.

Для каждой **Platform_Operation** таблица NIT_{actual} (и/или BAT в спутниковых сетях) содержит дескриптор ECI_ platform_operation_descriptor как минимум в центральном мультиплексе и таблице, определяемых в строке установки для сетей с ручной установкой и во всех мультиплексах для сетей с автоматическим обнаружением (кроме спутниковых сетей). Спутниковым сетям разрешено передавать дескриптор ECI_ platform_operation_descriptor только в мультиплексах, которые поставщик использует для предоставления услуг – как часть таблицы NIT или таблицы BAT.

ECI_ platform_operation _descriptor – это частный дескриптор_DVB, использующий спецификатор частных данных **ECI** в дескрипторе DVB private_data_specifier_descriptor [ETSI TS 101 162]. Он определяется в таблице 7.2.2.2-1.

Таблица 7.2.2.2-1 – ECI_ platform_operation _descriptor

Синтаксис	Количество битов	Мнемоника
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
/* цикл наименования платформы */		
для (i=0; i<N; i++){		
platform_name_char	8	uimsbf
}		
для (i=0; i<N; i++){		
extension_byte	8	uimsbf
}		
}		

Семантика

descriptor_tag	Значение маркера частных данных **ECI** для descriptor_tag: см. [b-ITU-T J Suppl. 7]
platform_tag	Данное 8-битовое поле определяет маркер **системы Platform_Operation** для целей ручной установки. В каждой таблице NIT и BAT сети каждая **система Platform_Operation** должна иметь уникальное значение platform_tag. Каждое значение platform_tag должно появляться в каждой таблице NIT или BAT только один раз. Значение platform_tag не должно использоваться для упорядочения поставщиков услуг и не должно быть представлено в интерфейсе **пользователя CPE** для выбора **системы Platform_Operation**
operator_id	Идентификатор оператора, определяемый в пункте 7.2.5 настоящей Рекомендации. Представляет собой идентификатор **оператора системы Platform_Operation**
platform_operation_id	Идентификатор **системы Platform_Operation**, определяемый в пункте 7.5.3 настоящей Рекомендации
platform_name_length	Длина последовательности октетов цикла наименования платформы. Если длина равна 0, поставщик услуг не поддерживает автоматическое обнаружение и не должен находиться в любом списке выбора поставщиков в меню установки клиента **CPE**. Максимальное значение данного поля равно 40
platform_name_char	Последовательность символов UTF8, представляющих название системы управления платформой
extension_byte	Дополнительные байты; зарезервированы в настоящей Рекомендации для использования в будущем

7.2.2.3 Дескриптор ECI_base_url_descriptor

Дескриптор ECI_base_url_descriptor позволяет **системе Platform_Operation** передавать базовый URL своего веб-интерфейса API (см. пункт 7.7.3), который может использоваться для предоставления услуг, связанных с установкой клиента в режиме онлайн-доступа.

Для каждой **системы Platform_Operation** таблица NIT_{actual} (и/или BAT в спутниковых сетях) может содержать ECI_base_url_descriptor в той же таблице, что содержит ECI_platform_operation_descriptor.

ECI_base_url_descriptor – это частный дескриптор DVB, использующий спецификатор частных данных ECI в дескрипторе DVB private_data_specifier_descriptor [ETSI EN 300 468]. Дескриптор определяется в таблице 7.2.2.3-1.

Таблица 7.2.2.3-1 – Дескриптор ECI_base_url_descriptor

Синтаксис	Количество битов	Мнемоника
ECI_base_url_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
platform_tag	4	uimsbf
reserved	4	
base_url_length	8	uimsbf
/* цикл базового url */		
для (i=0; i<N; i++){		
base_url_char	8	uimsbf
}		
}		

Семантика

descriptor_tag	Значение маркера частных данных **ECI** для descriptor_tag: см. [b-ITU-T J Suppl. 7]
platform_tag	Данное 4-битовое поле определяет маркер поставщика для целей ручной установки. В каждой таблице NIT и BAT сети каждая система **Platform_Operation** должна иметь уникальное значение platform_tag. Каждое значение platform_tag должно появляться в каждой таблице NIT или BAT только один раз. Значение platform_tag не должно использоваться для упорядочения систем **Platform_Operation** и не должно быть представлено в интерфейсе пользователя CPE для выбора системы Platform_Operation
base_url_length	Это поле указывает количество байтов в base URL loop
base_url_char	Последовательность символов UTF8, образующая базовый URL для системы управления платформой

7.2.2.4 Ручная установка

Система Platform_Operation может предоставить **пользователю** строку установки, которую **пользователь** может ввести в соответствующий пункт меню установки в интерфейсе **пользователя CPE** для установки **клиента ECI**. Строка установки определяется в соответствии с указаниями, приведенными в настоящем разделе. Строка установки является цифровым представлением двоичного числа переменной длины. Двоичное число в первом представлении старшего значащего бита может быть составлено путем конкатенации 3-битовых двоичных значений разрядов в данном представлении.

Это число представлено **пользователю** блоками по 4 разряда, и ввод данных в пользовательский интерфейс **CPE** также представлен блоками по 4 цифры.

Строка установки идентифицирует параметры, приведенные в таблице 7.2.2.4-1.

Таблица 7.2.2.4-1 – Параметры строки установки (количество битов)

Параметр	DVB-T/DVB-T2	DVB-C/DVB-C2	DVB-S/DVB-S2	IPTV	Мнемоника
Тип сети	3	3	3	3	uimsbf
Идентификатор сети	16	17	17	16	uimsbf
Маркер платформы	8	8	8	8	uimsbf
Маркер клиента	4	4	4	4	uimsbf
Заполнение	0	0	0	0	uimsbf
Контрольная сумма	5	5	5	5	uimsbf
Количество битов	36	36	36	36	uimsbf
Количество цифр	12	12	12	12	uimsbf
Количество блоков	3	3	3	3	uimsbf

Семантика

Network type 3-битовое поле. Значения типа сети приведены в таблице 7.2.2.4-2

Network ID Идентификатор DVB SI Table-id, содержащий ECI_service_provider_descriptor (см. пункт 7.2.2.2), который предоставляет подробную информацию, необходимую для услуг доступа, как указано в таблице 7.2.2.4-3

Platform tag 4-битовое поле, представляющее маркер поставщика для требуемого поставщика услуг в дескрипторе ECI_service_provider_descriptor в таблице NIT или BAT

Client tag 4-битовое поле, представляющее маркер поставщика для требуемого клиента в дескрипторе ECI_service_provider_descriptor, выбранного по маркеру поставщика в таблице NIT или BAT

Padding Поле длиной 0..2 бита со значением, равным 0, которое заполняет предыдущую строку до значения, кратного 3 битам

Checksum 5-битовое поле, образованное путем сложения последовательных 5-битовых блоков предыдущей строки. Последняя часть строки заполняется дополнительными нулевыми старшими разрядами до значения длиной 5 битов. Например, контрольная сумма строки 0b01011010 равна 0b01011 + 0x00010 = 0b01101. Используя контрольную сумму, **пользовательский** интерфейс **CPE** выявляет и отклоняет любые ошибочные данные, введенные **пользователем**

Таблица 7.2.2.4-2 – Представление значений типа сети

Тип сети	Значение
DVB-T/T2	0
DVB-C/C2	1
DVB-S/S2	2
IPTV	3
Зарезервировано	4..7

Таблица 7.2.2.4-3 – Представление идентификаторов сети

Тип сети	Значение идентификатора сети	Количество битов
DVB-C	0b0, за которым следует идентификатор сети таблицы NIT, или 0b1, за которым следует идентификатор BAT таблицы BAT	17
DVB-S/S2	0b0, за которым следует идентификатор сети таблицы NIT, или 0b1, за которым следует идентификатор BAT таблицы BAT	17 17

7.2.2.5 Установка с автоматическим обнаружением

При использовании данного метода установки оборудование **CPE** должно быть способно автоматически обнаруживать параметры сети, работающей на основе транспортного потока, и тем самым получать доступ ко всем транспортным потокам сети.

Каждая услуга в каждом из мультиплексов обозначается **маркером ECI Platform_Operations**, который может обеспечить доступ к услуге. Это может быть реализовано в таблице SDT для каждой услуги (см. пункт 7.2.2.6) либо в таблице NIT или BAT (только для спутниковых сетей) для каждого мультиплекса (см. пункт 7.2.2.6).

Оборудование **CPE** предлагает **пользователю** функцию установки любого **клиента ECI системы Platform_Operation** как составную часть процесса установки с автоматическим обнаружением. В случае если **пользователь** принимает решение устанавливать **клиента ECI системы Platform_Operation** в целях получения дешифрованных услуг через соответствующую сеть доступа, оборудование **CPE** по умолчанию устанавливает все **услуги**, отмеченные маркерами для данной **системы управления платформой** в центральном списке услуг оборудования **CPE**.

7.2.2.6 Дескриптор метки услуг ECI

Дескриптор ECI_service_tag_descriptor содержится в таблице SDT. Он маркирует каждую услугу от поставщиков услуг **ECI**, которые предлагают ее дескремблирование. Определение приводится в таблице 7.2.2.6-1.

Таблица 7.2.2.6-1 – Дескриптор метки услуг ECI

Синтаксис	Количество битов	Мнемоника
`ECI_service_tag_descriptor(){`		
`descriptor_tag`	8	uimsbf
`descriptor_length`	8	uimsbf
`platform_tag`	8	uimsbf
`}`		

Семантика

descriptor_tag	Значение маркера частных данных **ECI** для descriptor_tag: см. [b-ITU-T J Suppl. 7]
platform_tag	Данное значение platform_tag **системы ECI Platform_Operation**, указанное в списке в ECI_platform_operation_descriptor, содержится в таблице NIT или BAT сети

7.2.2.7 Дескриптор списка платформ ECI

Дескриптор списка платформы **ECI** предоставляет список **систем ECI Platform_Operation**, обеспечивающих доступ к услугам различных мультиплексов в сети. Дескриптор ECI_platform_list_descriptor содержится в таблице NIT и/или BAT. Определение приводится в таблице 7.2.2.7-1.

Таблица 7.2.2.7-1 – Дескриптор ECI_platform_list_desciptor

Синтаксис	Количество битов	Мнемоника
ECI_platform_list_descriptor(){		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
для (i=0;i<N;i++){		
platform_count	8	uimsbf
/* цикл платформ */		
для (j=1; j<M; j++){		
platform_tag	8	uimsbf
}		
service_count	16	uimsbf
/* цикл услуг */		
для (j=0; j<M; j++){		
service_id	16	uimsbf
}		
}		
}		

Семантика

descriptor_tag	Значение маркера частных данных ECI для descriptor_tag: см. в [b-ITU-T J Suppl. 7]
platform_count	8-битовое поле указывает количество маркеров поставщика в следующем цикле
platform_tag	Значение platform_tag системы ECI Platform_Operations, представленное в списке ECI_platform_operation_descriptor, который содержится в таблице NIT или BAT сети. Услуги, к которым относится обозначенная маркером система Platform_Operation, представлены далее в цикле услуг. Значения platform_tag могут появляться несколько раз во внешнем цикле данного дескриптора
service_count	16-битовое поле, представляющее количество идентификаторов service_id в следующем цикле
service_id	Идентификатор услуг DVB в мультиплексе таблицы NIT или BAT, к которой может быть получен доступ с использованием услуг доступа платформ, на которые приведена ссылка в предыдущем цикле платформ

7.2.3 Обнаружение клиентов в сетях на основе IP

7.2.3.1 Ручная установка

Оборудование **CPE**, имеющее доступ к IP-сетям, предлагает функцию ручного ввода URL в целях разрешения установки поставщика услуг. URL будет служить в качестве базового указателя для веб-интерфейса API.

ПРИМЕЧАНИЕ. – Оборудование **CPE** может предоставлять доступ к различным онлайн-услугам в качестве своих прикладных функций, часть из которых может быть загружена. Оборудование **CPE** может предоставлять API-интерфейс установки поставщика услуг и клиента, что позволяет автоматизировать процесс установки клиента для **пользователя**.

7.2.3.2 Установка на основе веб-страницы

Данное решение задачи по установке **клиента ECI** выходит за рамки настоящей Рекомендации и может быть рассмотрено в дополнительных спецификациях.

7.3 Хранение, проверка и активация

7.3.1 Общая политика обновлений

ECI поддерживает возобновляемость элементов, обеспечивая тем самым высокий уровень целостности данных. В связи с этим регулярно проверяется наличие обновлений для всех загружаемых элементов. Указанная политика загрузок и обновлений применима ко всем данным **клиентов ECI** и **систем управления платформой**, а также к сопутствующим данным аннулирования.

Хосты ECI предпринимают регулярные попытки проверки обновлений и информируют **пользователя** о необходимости тех или иных действий. Подробные требования, предъявляемые к политике обновлений, предложены в [b-ITU-T J Suppl. 7].

Хост ECI хранит **цепочку клиентов системы управления платформой** совместно с соответствующим **клиентом ECI**. Процедуры хранения и удаления выполняются в рамках установки и удаления **клиентов ECI**.

Хост ECI автоматически обновляет **сертификат** поставщика платформы и перезаписывает его более ранние версии.

7.3.2 Загрузка и хранение образа клиента ECI

Хост ECI, будучи одним из ресурсов управления **клиентом ECI**, сохраняет **образ клиента ECI**, необходимый для доступа к услугам или контенту, хранящимся в энергонезависимой памяти, только после выдачи **пользователем** разрешения (в неявной форме). Любая автоматизированная политика установки **клиентов ECI** предоставляет **пользователю** прозрачный метод управления ограничениями ресурсов, что позволяет обеспечить управление **клиентами ECI**, вполне понятное для **пользователя** и не приводящее к непредвиденной потере доступа к контенту или услугам. Соответственно любое удаление **образа клиента ECI** должно быть (в неявной форме) разрешено **пользователем**.

Для каждой **системы управления платформой хост ECI** хранит загружаемых **клиентов ECI** в энергонезависимой памяти совместно с их исходными идентификационными данными. Новые версии **клиента ECI** (включая только новые идентификационные данные) перезаписывают более ранние версии (для каждой **системы управления платформой**). Пример: если две **системы управления платформой** используют один и тот же тип **клиента ECI**, но разные версии, то **хост ECI** должен сохранить обе версии.

Минимальный размер образа, который оборудование **CPE** может хранить в каждом сегменте памяти **клиента ECI**, предложен в [b-ITU-T J Suppl. 7].

7.3.3 Подтверждение и активация клиента ECI

Хост ECI загружает последнюю (по номеру версии) **цепочку клиентов системы управления платформой** для **сертификата системы управления платформой** в **усовершенствованную систему безопасности** и предпринимает попытку установки открытого ключа **системы управления платформой** в соответствии с общими правилами обработки цепочек, как определяется в пункте 5.4.2.

Хост ECI загружает последнего **клиента ECI** в **усовершенствованную систему безопасности**. **Хост ECI** загружает совместную подпись клиента **системы управления платформой** в **усовершенствованную систему безопасности**. После этого, в соответствии с общими правилами обработки цепочек, определяемыми в пункте 5.5, **хост ECI** проводит подтверждение **клиента ECI** и проверяет подпись и совместную подпись для **образа клиента ECI**. Если происходит аннулирование, **хост ECI** должен уведомить **пользователя**.

Установка и активация нового **клиента ECI** происходят только в случае, если процесс подтверждения успешно завершен.

7.4 Форматы структуры цепочки клиентов ECI

7.4.1 Введение в форматы структуры цепочки клиентов ECI

На рисунке 7.4.1-1 изображена структура **цепочки сертификатов клиента ECI.** Цепочка начинается со **списка аннулирования** поставщика, затем следуют **сертификат поставщика систем безопасности** и **список аннулирования клиентов ECI**; завершается цепочка файлом **образа клиента ECI**. При наличии **серии образов** вводится дополнительный **сертификат образа клиента ECI**. Подпись клиента **системы управления платформой ECI** предоставляет для образа клиента вторую подпись, обеспечивающую приемлемость **клиента ECI** для системы управления платформой. Определение приводится в пункте 7.5.

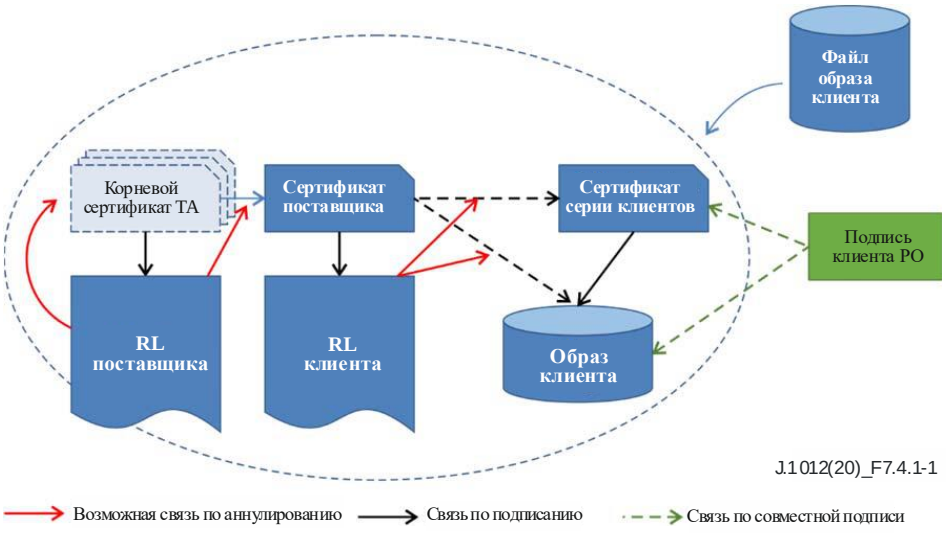


Рисунок7.4.1-1 – Цепочка аутентификации клиентов

7.4.2 Сертификат поставщика систем безопасности

Сертификаты поставщика систем безопасности определяются структурой ECI_Certificate. Идентификатор для **сертификата поставщика систем безопасности** определяется в таблице 7.4.2-1.

Таблица 7.4.2-1 – Определение идентификатора поставщика систем безопасности

Синтаксис	Количество битов	Мнемоника
`ECI_Vendor_Id {`		
`padding(4)`		
`type` `/* см. таблицу 5.2-2 */`	4	uimsbf
`vendor_id`	20	uimsbf
`vendor_version`	8	uimsbf
`}`		

Семантика

type: integer	Значение в соответствии с таблицей 5.2-2
vendor_id: integer	Номер поставщика, присвоенный поставщику систем безопасности, уникальный в контексте **ECI**
vendor_version: integer	Идентификатор, присваиваемый пошагово версии **сертификата** для поставщика систем безопасности. Значения 0x00 и 0xF0..0xFF резервируются

7.4.3 Сертификат серии клиента ECI и идентификатор назначения серии

Сертификаты серии **клиента ECI** определяются структурой **ECI_Certificate**. Идентификатор **сертификата поставщика систем безопасности** определяется в таблице 7.4.3-1.

Таблица 7.4.3-1 – Определение идентификатора серии клиентов

Синтаксис	Количество битов	Мнемоника
`ECI_Client_Series_Id {`		
`padding(4)`		
`type` `/* см. таблицу 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Семантика

type: integer	Значение в соответствии с таблицей 5.2-2
client_type: integer	**Тип клиента ECI**, уникальный в контексте идентификатора поставщика систем безопасности для **клиента ECI**
client_version_major: integer	Номер основной версии **клиента ECI** или **ECI Client-type**. Номера версий возрастают при выпуске нового основного варианта (см. примечание)
client_version_minor: integer	Номер дополнительной версии **клиента ECI. Клиенты ECI** могут быть аннулированы при сравнении номера дополнительной версии в **списках аннулирования клиентов ECI** и автоматически заменены
ПРИМЕЧАНИЕ. – Автоматическая замена **клиента ECI** при смене основного варианта в **ECI**-совместимых устройствах **CPE** не производится, поскольку лишь обновления дополнительных версии запускаются автоматически.	

ПРИМЕЧАНИЕ. – **Сертификаты** серии типа **клиента ECI** присваиваются **клиентам ECI**, которые требуют внедрения на своих условиях в отношении конкретного оборудования **CPE**, но при этом идентичны с точки зрения безопасности и функциональных возможностей.

Идентификатор назначения клиента определяется тем же способом, что и для **хостов ECI**, с использованием структуры ECI_Host_Series_Image_Target_Id. Тем самым образ клиента привязывается к определенному **хосту ECI**.

7.4.4 Подпись образа клиента ECI

Подписи **клиента ECI** используют структуру ECI_Data_Signature, как определяется в пункте 5.6.

Идентификатор **клиента ECI**, определяемый в таблице 7.4.4-1, идентичен по структуре идентификатору ECI_Client_Series_Id, как определяется в таблице 7.4.3-1.

Таблица 7.4.4-1 – Определение идентификатора клиента

Синтаксис	Количество битов	Мнемоника
`ECI_Client_Id {`		
`padding(4)`		
`type` `/* см. таблицу 5.2-2 */`	4	uimsbf
`client_type`	12	uimsbf
`client_version_major`	8	uimsbf
`client_version_minor`	8	uimsbf
`}`		

Семантика

type: integer	Значение в соответствии с таблицей 5.2-2
client_type: integer	Тип клиента, присвоенный **доверительным органом ECI**
client_version_major: integer	Номер основной версии **клиента ECI (ECI Client-type)**. Номера версий возрастают при выпуске нового основного варианта
client_version_minor: integer	Номер дополнительной версии **клиента ECI**. **Клиенты ECI** могут быть аннулированы при сравнении номера дополнительной версии в списках аннулирования **клиентов ECI**

7.5 Форматы цепочек системы управления платформой ECI

7.5.1 Обзор

На рисунке 7.5.1-1 представлена цепочка аутентификации для **сертификата системы управления платформой** и подписей клиента **системы управления платформой**. Она начинается со **списка аннулирования оператора**, за которым следуют **сертификат оператора, список аннулирования системы управления платформой** и, наконец, **сертификат системы управления платформой**, содержащий открытый ключ **системы управления платформой**. Эта цепочка используется совместно со **списком аннулирования** клиентов **системы управления платформой** для утверждения **образов клиентов ECI**, которым разрешено управление платформой.

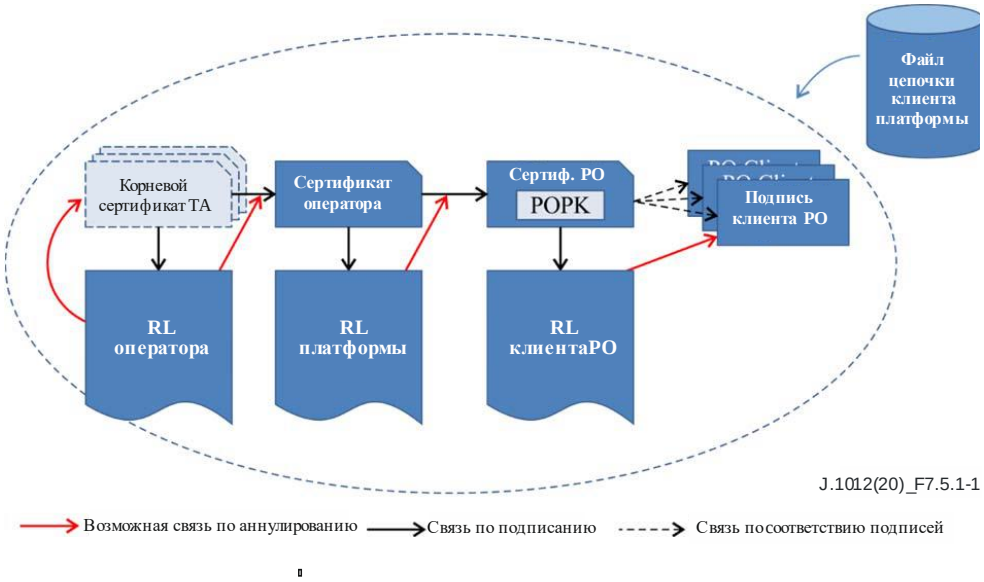


Рисунок 7.5.1-1 – Цепочка аутентификации для цепочки клиента платформы

7.5.2 Сертификат оператора

Сертификаты оператора определяются структурой **ECI_Certificate**. Идентификаторы для **оператора** определяются в таблице 7.5.2-1.

Таблица 7.5.2-1 – Определение идентификаторов оператора

Синтаксис	Количество битов	Мнемоника
ECI_Operator_Id {		
padding(4)		
type /* см. таблицу 5.2-2	4	uimsbf
operator_id	20	uimsbf
operator_version	8	uimsbf
}		

Семантика

type: byte	Значение в соответствии с таблицей 5.2-2
operator_id: integer	Присвоенный **оператору идентификатор**, уникальный в контексте корневого сертификата **ECI**
operator_version: integer	Номер версии, пошагово присваиваемый версии **сертификата** для **оператора**. Значения 0x00 и 0xF0..0xFF резервируются

7.5.3 Сертификат системы управления платформой

Сертификаты системы управления платформой определяются структурой **ECI_Certificate**. Секретным ключом для **системы управления платформой** распоряжается сама эта система. Идентификатор **сертификата для системы управления платформой** определяется в таблице 7.5.3-1.

Таблица 7.5.3-1 – Определение идентификатора системы управления платформой

Синтаксис	Количество битов	Мнемоника
ECI_Platform_Operation_Id {		
padding(4)		
type /* см. таблицу 5.2-2	4	uimsbf
platform_operation_id	20	uimsbf
platform_operation_version	8	uimsbf
}		

Семантика

type: byte	Значение в соответствии с таблицей 5.2-2
platform_operation_id: integer	Номер **системы управления платформой**, присвоенный поставщику систем безопасности, уникальный в контексте **сертификата** оператора
platform_operation_version: integer	Увеличивается в том случае, если **система управления платформой** изменяет свой **сертификат**

7.5.4 Список аннулирования клиентов системы управления платформой

Список аннулирования клиентов **системы управления платформой** определяется в пункте 5.3 с использованием присвоенного идентификатора, как указано в таблице 5.2-2. Поля entity_id fields в списке аннулирования ссылаются на поле cosignature_id структуры данных для подписи клиента **системы управления платформой**.

Номер минимальной версии списка аннулирования определяется в рамках инициализации **клиента ECI** и подтверждается с использованием усовершенствованной системы безопасности.

7.5.5 Совместная подпись клиента системы управления платформой

Совместная подпись клиента **системы управления платформой** предоставляет подпись **системы управления платформой** для проверки того, может ли быть выдано разрешение образу клиента на предоставление услуг доступа к платформе. Кроме того, подпись предоставляет идентификатор образа поставщика и клиента, упрощающий сопоставление с соответствующим образом клиента. Подписи клиентов **системы управления платформой** обладают собственной нумерацией идентификаторов; благодаря этому разрешено независимое аннулирование ранее разрешенных **образов клиента ECI** с использованием списка аннулирования клиентов **системы управления платформой**. Подробная информация приводится в таблице 7.5.5-1.

Таблица 7.5.5-1 – Определение совместной подписи системы управления платформой для клиента

Синтаксис	Количество битов	Мнемоника
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	
если (/* совместная подпись серий образов*/) {		
ECI_Client_Series_Id **client_series_id**	32	
format_version	8	uimsbf
если (format_version == 0x01){		
ECI_Signature_v1 **series_cosignature**		
}		
}		
если(/* совместная подпись образа */){		
ECI_Client_id **client_id**	32	
ECI_Data_Signature **image_cosignature**		
}		
}		

Семантика

type: byte	Значение в соответствии с таблицей 5.2-2
entity_id: integer	Уникальный **идентификатор**, присвоенный подписи в контексте **сертификата системы управления платформой**. Совместно с полем cosignature_version присваивается только одному **допустимому образу клиента**
version: integer	Значение возрастает (например, путем увеличения старших двоичных разрядов) в том случае, если **система управления платформой** изменяет свой открытый ключ. Младшие двоичные разряды поля могут использоваться для (частичного) представления версии **серии образов** клиента или образа клиента, с тем чтобы **системе управления платформой** было удобнее управлять аннулированием на основе версии клиента с использованием поля версии в списке аннулирования клиентов **системы управления платформой**
cosignature_id: ECI_PO_Cosignature_Id	Опознавание идентификатора совместной подписи на образе клиента. Данное поле включено в расчет совместной подписи
client_tag: integer	Короткая форма идентификатора, применяемая при установке для обозначения поля client_type в контексте **системы управления платформой**. Только клиенты, которые могут заменять друг друга с точки зрения **пользователя**, должны иметь одинаковое значение client_tag. Как правило, дополнительные версии клиента эквивалентны
vendor_id: ECI_Vendor_Id	Идентификатор **сертификата** поставщика для **образа клиента ECI**. Данное поле может использоваться для определения местоположения **серии образов** клиента или образа клиента, для которой в структуре данных представлена совместная подпись
client_series_id: ECI_Client_series_id	Идентификатор **сертификата** серии клиента для проверки образа. Тип поля client_series_id должен соответствовать полю child-type для client_image_series **сертификатов системы управления платформой** (см. таблицу 5.2-2); тем самым определяется правильный выбор альтернативных интерпретаций data-structure
format_version	Версия формата определения **сертификата**, применяемая для совместной подписи (см. таблицу 5.2-1), должна соответствовать определению версии **сертификата** клиента. Единственное заданное действительное значение данного поля равно 0x01
series_cosignature: ECI_Signature_v1	Совместная подпись сертификата client_image_series секретным ключом **системы управления платформой**. Данные, введенные в расчет подписи, определяются как идентичные сертификату client_image_series, заменяя client_image_series_id идентификатором cosignature_id этой структуры данных, а также заменяя поле расширения 4-байтовым расширением, содержащим исходное поле **сертификата** client_image_series_id
client_id: ECI_Client_Id	Идентификатор образа клиента. Тип поля client_id должен соответствовать полю child-type для client_image **сертификатов системы управления платформой**: см. таблицу 5.2-2; тем самым определяется правильный выбор альтернативных интерпретаций data-structure
image_cosignature: ECI_Data_Signature	Совместная подпись образа клиента секретным ключом **системы управления платформой**. Данные, введенные в расчет подписи, определяются следующим образом: за полем cosignature_id следуют данные в файле образа клиента, введенные в расчет подписи образа клиента, как определяется в пункте 7.6.1.

7.6 Форматы файлов

7.6.1 Формат файла образа клиента ECI

Файл идентификационных данных **клиента ECI** содержит данные, необходимые для проверки подлинности **клиента ECI доверительным органом ECI**. Используемый при этом формат определяется в таблице 7.6.1-1.

Таблица 7.6.1-1 – Определение идентификационных данных клиента

Синтаксис	Количество битов	Мнемоника
ECI_Client_Credentials {		
ECI_Certificate_Chain **client_chain**		
если (client_chain.chain_length == 0x1) { /* серия клиента отсутствует; типовой образ */		
ECI_RL **client_rl**		
}		
ECI_Data_Signature **client_signature**		
}		

Семантика

header: ECI_Client_Chain_Header	Заголовок файла цепочки **клиентов ECI**
client_chain: ECI_Client_Chain	**Цепочка сертификатов** для подтверждения **образа клиента ECI**, начинающаяся с корневого списка аннулирования поставщика систем безопасности и заканчивающаяся сертификатом поставщика систем безопасности для **клиентов ECI**, не связанных с сериями образов, или **сертификатом** серии **клиента ECI** для **клиентов ECI**, связанных с сериями образов
client_rl: ECI_RL	Список аннулирования для идентификаторов **образа клиента ECI**
client_signature: ECI_Data_Signature	Подпись для подтверждения **образа клиента ECI**; открытый ключ предоставляется **цепочкой клиента ECI**

Файл **образа клиента ECI** определяется в таблице 7.6.1-2.

Таблица 7.6.1-2 – Определение файла образа клиента ECI

Синтаксис	Количество битов	Мнемоника
ECI_Client_Image_File {		
magic = 'ECI'	24	uimsbf
image_header_version	8	uimsbf
ECI_Client_Credentials **credentials**		
если (image_header_version == 0x01) {		
если (credentials.client_chain.chain_length == 0x1) { /* типовой образ */		
ECI_Client_Id **client_id**	32	uimsbf
}		
если (credentials.client_chain..chain_length == 0x2) { /* Образ серии образов*/		
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	
для (i=0; i<n; i++) {		
client_image_byte	8	uimsbf
}		
}		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – три 8-битовых представления символов ECI в формате ASCII. **Хост ECI** проверяет значение данного поля, чтобы удостовериться, что формат файла **ECI** соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; все другие значения версий резервируются. **Хост ECI** игнорирует все образы, номер версии которых не распознается
credentials: ECI_Client_Credentials	**Идентификационные данные клиента ECI** для проверки подлинности **образа клиента ECI**
series_image: Boolean	Образ серии является не полем, а функцией, которая вычисляется на основе идентификационных данных, отображающих наличие **сертификата** серии типа **клиента ECI**
series_id: ECI_Client_Series_Id	Идентификатор серии **клиента ECI** серии образов для следующего образа. **Хост ECI** проверяет значение перед загрузкой **образа клиента ECI**
series_image_id: ECI_Client-series_Image_Id	Идентификатор образа в серии образов для следующего образа. **Хост ECI** проверяет значение перед загрузкой **образа клиента ECI**
client_id: ECI_Client_Id	Идентификатор **клиента ECI образа клиента ECI**. **Хост ECI** проверяет значение перед загрузкой **образа клиента ECI**
vendor_id: ECI_Vendor_Id	Идентификатор поставщика систем безопасности **образа клиента ECI**, определяемый в структуре ECI_Vendor_Id в пункте 7.4.2. **Хост ECI** проверяет данное поле перед загрузкой (нового) **образа клиента ECI**
image_encrypted_flag: integer	Этот флаг сигнализирует о том, шифруется ли образ. Если значение данного поля равно 0b0, то образ не шифруется. Если значение данного поля равно 0b1, то образ шифруется
online_flag: integer	Данный флаг сигнализирует о том, требуется ли протоколу онлайн-взаимодействие с сервером инициализации при использовании одноразового кода для получения ключа в целях дешифрования образа. См. пункт 7.8.3
client_image_byte: byte	Последовательность байтов, содержащая образ клиента

В таблице 7.6.1-2 фраза "**хост ECI** проверяет" означает, что **хост ECI** должен проверять соответствие ожидаемых значений значению в поле.

Подпись **образа клиента ECI** должна быть рассчитана по всем данным в файле, следующим за полем идентификационных данных.

7.6.2 Данные цепочки по управлению платформой

Файл **образа клиента ECI** определяется в таблице 7.6.2-1.

Таблица 7.6.2-1 – Определение файла цепочки по управлению платформой

Синтаксис	Количество битов	Мнемоника
ECI_Operation_Certificate_File {		
magic = 'EPC'	24	uimsbf
version	8	uimsbf
если (version == 0x01) {		
ECI_Certificate_Chain **operation_chain**		
ECI_RL po_client_rl		
client_image_count	16	uimsbf
для (i=0; i<client_image_count; i++) {		
ECI_PO_Client_Cosignature_Data **po_client_data**		
}		
ECI_RL po_client_rl		
}		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – это три 8-битовых представления символов 'EPC' в формате ASCII. **Хост ECI** проверяет значение данного поля, чтобы удостовериться, что формат файла **ECI** соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
Image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; все другие значения версий резервируются. **Хост ECI** игнорирует все образы, номер версии которых не распознается
operation_chain: ECI_Client_Chain	**Цепочка сертификатов** для подтверждения **образа клиента ECI**, начинающаяся с корневого списка аннулирования оператора и заканчивающаяся сертификатом **системы управления платформой**
po_client_rl: ECI_RL	Список аннулирования клиентов **системы управления платформой**, используемый для подтверждения совместных подписей образа клиента. **Хост ECI** проверяет cosignature_ids в po_client_data в рамках проверки совместной подписи
client_image_count: integer	Количество структур данных подписи для образов клиентов в следующем цикле

В таблице 7.6.2-1 фраза "**хост ECI** проверяет" означает, что **хост ECI** должен проверять соответствие ожидаемых значений значению в поле.

7.6.3 Файлы данных аннулирования

Существуют два типа файлов данных аннулирования для **загрузчика клиента ECI**. Оба файла используют формат ECI_Revocation_Data_File, определяемый в таблице 5.5-2.

Файл данных аннулирования **клиентов ECI** использует значение father_type, равное 0x0 (**корневой сертификат**), и значение sub_type, равное типу списка аннулирования поставщика. Revocation_data имеет следующее ограничение: списки аннулирования конечных узлов в деревьях являются списками аннулирования **клиентов ECI**.

Файл данных аннулирования **системы управления платформой** использует значение father_type, равное 0x0 (**корневой сертификат**), и значение sub_type, равное типу списка аннулирования **оператора**. Revocation_data имеет следующее ограничение: списки аннулирования конечных узлов в деревьях являются списками аннулирования **системы управления платформой**.

7.7 Транспортные протоколы ресурсов клиента ECI

7.7.1 Общие сведения и профилирование

В настоящем разделе определяется применение протоколов в оборудовании **CPE** и **системах управления платформой**.

Радиовещательный протокол не обеспечивает работу с **сериями образов**. Образы на основе серий предусматриваются только для устройств, подключенных по IP-протоколу.

Оборудование **CPE**, поддерживающее как радиовещательный, так и онлайн-доступ к ресурсам **клиента ECI**, использует радиовещательный доступ с более высоким приоритетом (если не указано иное в настоящей Рекомендации), чтобы разгрузить онлайн-трафик. Однако онлайн-доступ может использоваться в срочных случаях (ожидание со стороны **пользователя**) и должен использоваться в том случае, если радиовещательная сеть не обеспечивает минимально необходимых частот для доступа.

7.7.2 Радиовещательный транспортный протокол

7.7.2.1 Введение

Для обеспечения возможности инициализации и поддержки **клиента ECI** интерфейсу **ECI** требуются дополнительные данные для различных функций со стороны **клиента ECI** и/или **хоста ECI**. Для всех типов данных используется один и тот же транспортный протокол, определяемый в настоящем разделе. Он тесно связан с протоколом, используемым для загрузки файлов **образа хоста ECI**.

Для передачи по каналам радиовещания данные разбиваются на блоки при помощи хеш-функции на основе индекса доступа, используемого оборудованием **CPE** для определения необходимости получения данных. Использование блоков позволяет значительно сократить объем данных, которые

должно загрузить оборудование **CPE**, и повысить избирательность отслеживания изменений в данных, которые действительно необходимы оборудованию **CPE**.

Определяются (по типу контента) следующие отдельные группы каруселей:

- образы клиента ECI (для каждого поставщика систем безопасности);
- данные аннулирования **клиентов ECI**, структурированные в блоки на основе индексов <client_id, client-version_major> и vendor_id;
- **цепочка сертификатов** системы управления платформой;
- данные аннулирования **системы управления платформой**, структурированные в блоки на основе индексов provider_id и operator_id;
- данные аннулирования **образа хоста ECI**, структурированные в блоки;
- данные инициализации **клиента ECI** AS_setup **ECI**, структурированные в блоки;
- для структуры данных импорта и экспорта определяются группы каруселей (см. пункт 9.8);
- для проприетарных данных **оператора** определяются группы каруселей.

Все параметры карусели DSMCC должны соответствовать [ETSI EN 301 192].

Для передачи всех необходимых данных **оператор** может использовать несколько каруселей в отдельных мультиплексах. Однако для любого конкретного **клиента ECI хост ECI** должен лишь контролировать обновления отдельного DII расположения карусели данных.

7.7.2.2 Передача оператору в режиме хендовера идентификационных данных и данных аннулирования

Форматы данных и протоколы для передачи **оператору** идентификационных данных и списков аннулирования не являются частью спецификации интерфейса **ECI**.

7.7.2.3 Передача данных в режиме хэндовера от поставщика системы безопасности оператору

Форматы данных и протоколы для передачи контента от **поставщика систем безопасности оператору** не входят в состав настоящей Рекомендации.

7.7.2.4 Передача сигналов PSI

Карусели должны использовать дескриптор stream_identifier_descriptor [ETSI EN 300 468] в таблице PMT для маркировки потока, используемого для передачи карусели, с тем чтобы разрешить создание ссылок дескриптором data_broadcast в SI.

Карусели должны использовать дескриптор data_broadcast_id_descriptor с идентификатором data_broadcast_id, как определяется в таблице 7.7.2.4-1.

Таблица 7.7.2.4-1 – Значение идентификатора радиовещательной передачи данных для отдельных каруселей ECI

Значение data_broadcast_id	Значение
Распределяется проектным бюро DVB, см. значение broadcast-id, определяемое в[ETSI TS 101 162].	Конкретная карусель передачи дополнительных данных клиента для **оператора ECI**

Байты селектора дескриптора data_broadcast_id_descriptor должны соответствовать структуре, определяемой в таблице 7.7.2.4-2.

Таблица 7.7.2.4-2 – Структура идентификатора карусели для каруселей передачи данных ECI DVB DSMCC

Синтаксис	Количество битов	Мнемоника
ECI_carousel_id_structure {		
version	8	uimsbf
если (version == 0x01){		
operator_id	20	uimsbf
platform_**operation_id**	20	uimsbf
}		
}		

Семантика

version: integer	Версия структуры; на данный момент определяется только одно значение 0x01. Все другие значения резервируются. Если какая-либо версия отличается от 0x01, устройства **CPE** должны игнорировать данный дескриптор.
operator_id: ECI_Operator_Id	Идентификатор **ECI оператора** (который определяется для любого **сертификата оператора**) **системы управления платформой**, относящейся к карусели
platform_operation_id: ECI_Platform_Operation-Id	В соответствии с **сертификатом системы управления платформой** – ID **системы управления платформой**

7.7.2.5 Передача сигналов SI

7.7.2.5.1 Передача сигналов о местоположении карусели данных через дескриптор связи для определения местоположения данных

Дескриптор связи для определения местоположения данных **клиента ECI** является частным дескриптором связи DVB **ECI** [ETSI TS 101 162]. Данный дескриптор связи помогает оборудованию **CPE** определить местоположение мультиплекса, передающего карусель данных **клиента ECI** для конкретной **системы управления платформой**. Данный дескриптор связи содержится в таблицах NIT или BAT. Дескриптору связи для определения местоположения данных **клиента ECI** в компоненте таблицы всегда предшествует дескриптор спецификатора частных данных DVB [ETSI TS 101 162] со значением поля private_data_specifier, равным "ECI", как определяется в [ETSI TS 101 162]. Этот дескриптор может появляться в таблицах NIT или BAT неоднократно. Данный дескриптор связи содержится в сетях и пакетах, в которых работают более четырех мультиплексов.

Согласно определению дескриптора связи, приведенному в [ETSI EN 300 468] и [ETSI TS 101 211], поля дескриптора связи для определения местоположения данных **клиента ECI** имеют следующее конкретное применение:

- **service_id –** может быть задано равным 0x0000, что означает отсутствие информации о конкретном идентификаторе service_id;
- **linkage_type –** значение 0x80 передает информацию о дескрипторе связи для определения местоположения данных клиента **ECI**.

Поле байтов частных данных дескриптора связи для установления местоположения данных **клиента ECI** содержит структуру, определяемую в таблице 7.7.2.5.1-1.

Таблица 7.7.2.5.1-1 – Структура частных данных для дескриптора связи по определению расположения карусели передачи данных клиента ECI

Синтаксис	Количество битов	Мнемоника
ECI_client_data_location {		
version	8	uimsbf
если (version==0x01){		
для (i=0;i<n; i++){		
operator_id	20	uimsbf
platform_**operation_id**	20	uimsbf
}		
}		
}		

Семантика

version: integer	Версия структуры; на данный момент определяется только значение 0x01. Все другие значения резервируются. Если какая-либо версия отличается от 0x01, устройства **CPE** должны игнорировать данный дескриптор
operator_id: ECI_Operator_Id	Идентификатор **ECI оператора** (который определяется **сертификатом оператора**) **системы управления платформой**, относящейся к карусели. Значение 0x00000 означает любого **оператора**
platform_operation_id: ECI_Platform_Operation-Id	В соответствии с **сертификатом системы управления платформой –** ID **системы управления платформой**. Значение 0x0000 означает любую **систему управления платформой**

Операторы сетей и поставщики пакетов услуг могут использовать шаблонные спецификаторы (значение 0x00000) для идентификаторов operator_id или platform_operation_id в целях связи с мультиплексом, который содержит одну или несколько каруселей передачи данных **клиента ECI**. В целях повышения эффективности рекомендуется, чтобы такая передача сигналов ограничивалась содействием устройствам **CPE** в рассмотрении вопроса о минимальном количестве мультиплексов, необходимом для определения расположения конкретной карусели **системы управления платформой**.

Рекомендуется, чтобы в таблицах NIT или BAT применялся только один дескриптор связи для определения местоположения карусели передачи данных **клиента ECI** к мультиплексу и чтобы все подходящие карусели, расположенные в данном мультиплексе, были указаны в одной структуре ECI_Client_data_location.

7.7.2.5.2 Дескриптор срочной загрузки клиента ECI

Чтобы указать на необходимость срочной замены **образа клиента ECI**, один или несколько дескрипторов ECI_client_emergency_download могут быть помещены в таблицы NIT, BAT или в одну из записей SDT для услуги, доступ к которой может обеспечить отмеченный **клиент ECI**. **Хост ECI** должен быть способен в течение не более чем 30 минут извлечь этот дескриптор из любой таблицы, в которой он появляется, в любом из настроенных мультиплексов и выполнить соответствующую обработку, а также использовать любое резервное устройство настройки для доступа к надлежащим мультиплексам в целях получения данного дескриптора.

Дескриптор ECI_client_emergency_download_descriptor разрешает назначать целевыми объектами конкретные системы управления платформами и конкретные типы хостов в целях минимизации нарушений в работе, вызванных срочными обновлениями.

Когда **хост ECI** обнаруживает новый дескриптор ECI_client_emergency_download (проверенный по table-origin и полю emergency_id), он должен соответствовать собственной конфигурации хоста ECI и клиента ECI в отношении содержащейся с дескрипторе информации по намеченным целям. Если обнаружено совпадение целевого объекта и версия установленного на данный момент образа клиента требует обновления, хост должен выполнить это обновление в соответствии с индикатором emergency_indicator. Это может вызвать нарушение в работе **пользователей** на оборудовании **CPE** в случае конфликтов ресурсов.

Операционный дескриптор **ECI** является частным дескриптором DVB; и в таблице, в которой он появляется, ему всегда предшествует дескриптор DVB private_data_specifier_descriptor с использованием поля **ECI** private_data_specifier_field (см. [ETSI EN 300 468]). Синтаксис дескриптора определяется в таблице 7.7.2.5.2-1.

Таблица 7.7.2.5.2-1 – Дескриптор ECI_Client_Emergency_Download_Descriptor

Синтаксис	Количество битов	Мнемоника
ECI_client_emergency_download_descriptor{		
descriptor_tag	8	uimsbf
descriptor_length	8	uimsbf
/* основной цикл */		
main_loop_nr	8	uimsbf
для (i=0; i<main_loop_nr; i++){		
/* целевая платформа */		
platform_operation_tag	8	uimsbf
/* цикл намеченной задачи хоста */		
host_nr	8	uimsbf
/* цикл намеченной задачи **id** хоста ***/**		
для (j=0; j<host_nr; j++){		
manufacturer_id	20	uimsbf
cpe_type_id	20	uimsbf
host_version	8	uimsbf
}		
/* цикл образа клиента */		
client_nr		
для (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
}		
}		
/* частные данные до конца дескриптора*/		
для (i=0; i<n; i++){		
private_data_byte	8	
}		
}		

Семантика

descriptor_tag	Значение маркера частных данных **ECI** для descriptor_tag: см. [b-ITU-T J Suppl. 7]
descriptor_length	См. [ETSI EN 300 468]
main_loop_nr	Количество записей в основном цикле. Отдельные записи основного цикла должны оцениваться **хостом ECI** по отдельности, то есть обладать семантикой OR. Различные элементы одной записи цикла должны обладать семантикой AND.
platform_operation_tag	Значения маркеров для платформы **ECI**, перечисленные в дескрипторе ECI_platform_operation_descriptor в таблице NIT/BAT. **Хост ECI** должен предусматривать срочное обновление, если platform_operation соответствует platform_operation одного из установленных **клиентов ECI**
host_nr	Количество записей в цикле намеченной задачи хоста; значение 0 означает, что все **хосты ECI** являются целевыми объектами. Записи цикла должны обладать семантикой OR; то есть если какая-либо спецификация намеченной задачи хоста соответствует условию намеченной задачи, основной цикл находится в состоянии соответствия
manufacturer_id	Идентификатор manufacturer_id хоста, для которого предназначено срочное обновление. **Хост ECI** должен предусматривать срочное обновление, если значение данного поля соответствует manufacturer_id хоста
cpe_type_id	Значение, определяемое идентификатором ECI_CPE_Type_ID в таблице 6.2.2.1-2. **Хост ECI** должен предусматривать срочное обновление, если значение поля cpe_type_id хоста соответствует значению данного поля. Значение cpe_type_id.cpe_type, равное 0x000, означает соответствие любого cpe_types **хоста ECI** (а cpe_model и host-version следует игнорировать). Значение cpe_type_id.cpe_model, равное 0x00, означает соответствие любого cpe_model **хоста ECI** (а версию хоста следует игнорировать)
host_version	**Хост ECI** должен предусматривать срочное обновление только в том случае, если его версия меньше или равна значению данного поля. См. примечание
client_nr	Количество записей в цикле образа клиента. Записи цикла должны обладать семантикой OR, и все соответствующие образы клиентов должны учитываться при срочном обновлении
emergency_indicator	**Хост ECI** использует значение данного поля в целях выбора соответствующего режима для начала загрузки и последующего обновления клиента, как указано в таблице 7.7.2.5.2-2
client_tag	Значение маркера, идентифицирующее **клиента ECI**, как указывается в дескрипторе ECI_platform_operation_descriptor в таблице NIT/BAT, которое соответствует полю platform_operation_tag в том же основном цикле. **Хост ECI** должен предусматривать срочное обновление, если рассматриваемые идентификаторы vendor_id и client_id соответствуют одному из установленных клиентов в **хосте ECI**

min_client_version_major	В данном поле представлен минимально допустимый номер основной версии для образа клиента. **Хост ECI** должен предусматривать срочное обновление, если устанавливается клиент, соответствующий client_tag, основная версия которого меньше значения данного поля
min_client_version_minor	В данном поле представлен минимально допустимый номер дополнительной версии для образа клиента. **Хост ECI** должен предусматривать срочное обновление, если устанавливается **клиент ECI**, соответствующий client_tag, дополнительная версия которого меньше значения данного поля, а основная версия равна min_client_version_major
client_id	Идентификатор **клиента ECI**, который обеспечивает услуги дешифрования для услуг с маркером platform_operation_tag, как определено в таблице 7.4.4-1
private_data_byte	Частные данные: контент может определяться **оператором**, управляющим вещательной передачей этого дескриптора
ПРИМЕЧАНИЕ. – Значение поля, равное 0xFF, предполагает соответствие всех версий хоста.	

В таблице 7.7.2.5.2-1 определяется ряд условий в основном цикле (семантика AND), которые должны соблюдаться, чтобы **хост ECI** смог выполнить срочное обновление. Если все эти условия выполнены, **хост ECI** осуществляет срочную загрузку и установку одного или нескольких образов клиента в соответствии с полем emergency_indicator для данного клиента.

Таблица 7.7.2.5.2-2 – Значения поля ECI_Client_emergency_download_descriptor emergency_indicator

Название	Значение	Описание
Экстренное системное обновление	0x01	**Хост ECI** загружает новый образ клиента и устанавливает как можно быстрее. При этом в случае необходимости допускается прерывание текущего сеанса работы **пользователя** (см. примечание 1)
Экстренное обновление клиента	0x02	**Хост ECI** загружает новый образ клиента и устанавливает его перед любым открытием сеанса работы под управлением медийных средств для данного клиента. Но сначала все текущие сеансы работы под управлением медийных средств для данного клиента должны быть завершены (см. примечание 2)
Срочное обновление клиента	0x03	**Хост ECI** загружает новый образ клиента и устанавливает его при первой возможности, не нарушая текущей работы **пользователя. Хост ECI** загружает новый образ хоста самое позднее в ходе следующего включения оборудования (см. примечание 3)
RFU	Прочее	Зарезервировано для использования в будущем
ПРИМЕЧАНИЕ 1. – Этот сценарий может использоваться **операторами**, например в том случае, если действующий **клиент ECI** может нанести вред **хосту ECI** и/или другим **клиентам ECI** и должен быть заменен немедленно. ПРИМЕЧАНИЕ 2. – Этот сценарий может использоваться **операторами**, например в том случае, если характеристики действующего **клиента ECI** недостаточно эффективны для дешифрования **услуг**. ПРИМЕЧАНИЕ 3. – Этот сценарий может использоваться **операторами**, например в том случае, если действующий **клиент ECI** испытывает серьезный недостаток средств дешифрования, однако может функционировать надлежащим образом в стандартных сценариях использования.		

7.7.2.6 Дескриптор совместимости карусели

Дескриптор compatibilityDescriptor, применяемый в карусели передачи данных [ETSI EN 301 192], используется в сообщениях DSI DII.

Дескриптор compatibilitydescriptor предоставляет информацию о типе данных, передаваемых в группе каруселей. SpecifierData() содержит идентификатор OUI **ECI**. В таблице 7.7.2.6-1 определяются действующие поля compatibilityDescriptor в каруселях передачи данных **клиента ECI**.

Таблица 7.7.2.6-1 – Типы контента карусели передачи данных ECI

Поле типа дескриптора	Групповое назначение	Поле модели	Поле версии	Индекс блока данных для расчета ID модуля
0xA0	**Образы клиентов ECI** и файлы идентификационных данных для одного поставщика	Vendor_id **поставщика систем безопасности** образов		Присваивается без ограничений
0xA2	Файлы данных аннулирования **клиентов ECI** (в виде блоков данных)	platform_operation_id		= Vendor_id + <Client_type, client_version_major> (см. примечание)
0xA3	Файл цепочки **системы управления платформой**	platform_operation_id, platform_operation_version		Присваивается без ограничений
0xA4	Файлы данных аннулирования **системы управления платформой** (в виде блоков данных)	platform_operation_id		= Operator_id + provider_id
0xA5	Файлы данных аннулирования **хоста ECI** (в виде блоков данных)	platform_operation_id		= Manufacturer_id + cpe_type_id
0xA6	Файлы AS_setup (в виде блоков данных)	platform_operation_id		target_id для **CPE**
0xA7-0xAA	Контейнер приложения UI (см. пункт 9.4.3.4.2)	Определяется оператором		Присваивается без ограничений
0xB0	Файл дерева экспорта	platform_operation_id (экспорта **клиента ECI**)		Присваивается без ограничений
0xB1	Файл цепочки импорта	platform_operation_id (импорта **клиента ECI**)		Присваивается без ограничений
0xB2	Файл цепочек аутентификации импорта	platform_operation_id (импорта **клиента ECI**)		Присваивается без ограничений
0xB8-0xBF	Проприетарный формат оператора	Определяется оператором		Определяется оператором
Прочие значения	Зарезервировано			
ПРИМЕЧАНИЕ. – Конкатенация двух полей (старшее значащее поле служит первым аргументом) с образованием 20-битового числа.				

Расчет индекса блока данных производится с использованием 32-битовой модулярной целочисленной арифметики и описывается в пункте 7.7.2.7.

7.7.2.7 DSI карусели

В случае применения двухуровневой карусели в DSI должен быть представлен полный индекс групп в карусели (то есть одна запись цикла на каждое сообщение DII).

Дескриптор compatibilityDescriptor определяется в таблице 7.7.2.6-1. Поля DII, не связанные с циклом, применяются со следующими ограничениями:

- размер блока (block Size) должен быть равен как минимум 512 байтам; для групп с более длинными модулями рекомендуется размер не менее 2 кбайтов;
- поле tCDownloadScenario – значение должно быть как минимум в 4 раза выше времени повтора наиболее медленного сообщения DDB в группе. Поле TCDownload также должно применяться с максимальными ограничениями, определенными в таблице B.4-1;
- поле numberOfModules отражает количество модулей для стандартных каруселей и количество блоков данных (каждый соответствует одному модулю) для сгруппированных данных. Значение для данных **цепочки сертификатов системы управления платформой** должно составлять 1.

Приведенные ниже значения поля tCDownloadScenario отражают период времени ожидания для получения всех элементов данных оборудованием **CPE**. Значение должно быть как минимум в четыре раза выше времени повтора наиболее медленного сообщения DDB в любом из модулей группы. Значения различных элементов определяются в разделе B.4.

Ниже приведены поля цикла модуля, применяемые со следующими ограничениями:

- поле moduleId – биты с 15 по 8 должны быть такими же, как и младший двоичный разряд идентификатора groupId в соответствующей структуре groupInfo в DSI. Биты с 7 по 0 присваиваются в соответствии с таблицей 7.7.2.7-1;
- moduleVersion – применение зависит от типа карусели; поле должно соответствовать таблице 7.7.2.7-1;
- moduleInfoLength – 0 для всех каруселей **ECI**.

Таблица 7.7.2.7-1 – Параметры группы карусели ECI

Тип группы	ModuleId, биты 7..0	ModuleVersion	ModuleInfo
Образы клиентов	client_type	client_version	Нет
Данные аннулирования клиентов	bucket_number	Увеличивается при каждом обновлении	Нет
Цепочка клиентов **системы управления платформой**	Присваивается **оператором**	Увеличивается при каждом обновлении	Нет
Данные аннулирования **системы управления платформой**	bucket_number	Увеличивается при каждом обновлении	Нет
Данные аннулирования **хоста ECI**	bucket_number	Увеличивается при каждом обновлении	Нет
Данные AS_setup ECI	bucket_number	Увеличивается при каждом обновлении	Нет

Для сгруппированного числа номер блока (равный биту [7..0] индикатора module_id) должен вычисляться на основе индекса путем простой операции по модулю:

`bucket_number = bucket Index %` numberOfModules

7.7.2.8 DDB карусели

Конкретные требования отсутствуют.

7.7.2.9 Режим динамической карусели

Нумерация версий карусели, а также обновления DSI и DII должны соответствовать [ETSI TR 101 202]. Это предполагает, что любое обновление модуля должно быть отражено в номере версии модуля, его DII и далее последовательно до DSI (при наличии).

Для выполнения всех динамических обновлений в нормальном режиме работы введенное в действие оборудование **CPE** отслеживает изменения в своих целевых модулях.

7.7.3 Транспортные веб-протоколы

7.7.3.1 Введение

Необходимые элементы данных могут быть получены **хостом ECI** от сервера, назначаемого **оператором**.

Интерфейс использует прямые HTTPS-запросы, как указывает в пункте 9.4.4.6, и следует принципам разработки RESTfull [b-Richardson]; при этом запрос кодируется как комбинация расширения URL и параметров запроса, а отклик кодируется как двоичный файл.

Сервер HTTP отвечает посредством одного из следующих кодов состояния:

- 200 – OK (запрашиваемый файл возвращается);
- 302 FOUND – запрос перенаправляется на другой сервер с отсрочкой; http-запрос повторяется в соответствии с возвращенным URL;
- 404 – элемент отсутствует на сервере;
- 500.. 599 – ошибка сервера.

В спецификации адресов URL, используемых для запросов, применяется спецификация формы Бэкуса–Наура. Названия символов, соответствующих полям в структурах данных **ECI**, приводятся в шестнадцатеричном представлении (строка символов '0'.. '9', 'A'.. 'F'); при этом количество разрядов в два раза превышает количество байтов, используемых для представления числа во внутренних

структурах двоичных данных **ECI**. Сервер игнорирует любые дополнительные параметры запроса, которые он не распознает.

7.7.3.2 Обзор веб-интерфейса API ECI

Оператор поддерживает онлайн-сервер, который откликается на запрос HTTP1.1 [IETF RFC 7231] GET, соответствующий приведенным ниже синтаксису и семантике URL:

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

Обозначения **major** и **minor** означают номера основной и дополнительной версий протокола в десятичном виде без предваряющих нулей. Текущая версия имеет номер 1.0. Определение концевой комбинации дается в таблице 7.7.3.2-1.

Таблица 7.7.3.2-1 – Определение концевой комбинации

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

Tail_extension обозначает различные варианты расширения для веб-интерфейса API **ECI**, как определяется в настоящей Рекомендации.

7.7.3.3 Запросы веб-интерфейса API, относящиеся к хосту ECI

Следующие запросы веб-интерфейса API, относящиеся к **хосту ECI**, определяются ниже.

- host_version::= 'host-version ' '?target-id=' target_id.
  Это выражение возвращает последнюю версию набора **образов хоста ECI** для оборудования **CPE**, идентифицируемого по **target_id**.
- host_images::= 'hi-images' ' ?target-id=' target_id.
  Это выражение возвращает последнее количество образов **хоста ECI** для оборудования **CPE**, идентифицируемого по **target_id**.
- host_image_version::= 'hi-version' ' ?target-id=' target_id '&image-id=' image_id.
  Это выражение возвращает последнюю версию **image_id** файла **образа хоста ECI** для оборудования **CPE**, идентифицируемого по **target_id**.
- host_image::= 'host-image' ' ?target-id=' target_id '&image-id=' image_id.
  Это выражение возвращает последнее значение **image_number образа хоста ECI** для оборудования **CPE**, идентифицируемого по **target_id**. Выражение image_number=="FF" возвращает файл идентификационных данных **хоста ECI** для **образов хоста ECI**, в том числе последние данные аннулирования.

Для запросов, связанных с **хостом ECI**, сервер **системы управления платформой** может поддерживать **хосты ECI** для любого типа оборудования **CPE** на выбор. Если сервер поддерживает какой-либо тип оборудования **CPE**, он также должен поддерживать полный последний набор **образов хоста ECI** и соответствующие запросы **host_image_version**, **host_images** и **host_revocation**. ECI_Host_Version_File представляет собой формат возвращенного файла, как определяется в таблице 7.7.3.3-1.

Таблица 7.7.3.3-1 – Определение файла версии хоста ECI

Синтаксис	Количество битов	Мнемоника
ECI_Host_Version_File {		
magic = 'RHVE'	32	uimsbf
host_version	8	uimsbf
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RHIM' в формате ASCII
host_version: integer	Номер версии **сертификата хоста ECI**

Возвращаемый файл имеет формат ECI_Host_Images_File, как определяется в таблице 7.7.3.3-2.

Таблица 7.7.3.3-2 – Определение файла образов хоста

Синтаксис	Количество битов	Мнемоника
ECI_Host_Images_File {		
magic = 'RHIM'	32	uimsbf
host_images	8	uimsbf
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RHIM' в формате ASCII
host_images: integer	Количество **образов хоста ECI,** поддерживаемое оборудованием **CPE,** тип которого определен в запросе.

Возвращаемый файл имеет формат ECI_Host_Image_Version_File, как определяется в таблице 7.7.3.3-3

Таблица 7.7.3.3-3 – Синтаксис файла версии образа хоста

Синтаксис	Количество битов	Мнемоника
ECI_Host_Image_Version_File {		
magic = 'RHIV'	32	uimsbf
host_image_version	16	uimsbf
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RHIV' в формате ASCII
host_image_version: integer	Версия **образа хоста ECI,** определяемая запросом

7.7.3.4 Запросы веб-интерфейса API, относящиеся к системе управления платформой

Сервер **системы управления платформой** должен поддерживать следующие запросы относительно идентификаторов **системы управления платформой**, которые он поддерживает:

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

Это выражение возвращает состояние аннулирования сертификата, выпущенного для идентификаторов **operator_id**, **platform_operation_id** в формате файла, определяемом в таблице 7.7.3.4-1. При функционировании через данный интерфейс сервер **системы управления платформой** должен как минимум поддерживать собственные **сертификаты системы управления платформой**:

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

Это выражение возвращает состояние аннулирования платформы **образа клиента ECI** для идентификатора cosignature_id в соответствии с последним списком аннулирования клиентов системы управления платформой. См. таблицу 7.7.3.4-2:

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

Это выражение возвращает последнюю цепочку **клиентов ECI** для **системы управления платформой**, которая опознается по идентификаторам **operator_id**, **platform_operation_id**, как определяется в таблице 7.6.2-1. При функционировании через данный интерфейс сервер **системы управления платформой** должен как минимум поддерживать собственные **сертификаты системы управления платформой:**

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

Это выражение возвращает последний файл данных аннулирования **системы управления платформой**, содержащий список аннулирования для **оператора**, который опознается по идентификатору **operator_id**. Сервер должен, по меньшей мере, поддерживать последние данные аннулирования для **оператора** его собственной **системы управления платформой**. **Хосты ECI** используют этот API для попыток получения последних данных аннулирования всех сохраняемых **клиентов ECI**.

Таблица 7.7.3.4-1 – Синтаксис файла проверки системы управления платформой

Синтаксис	Количество битов	Мнемоника
ECI_PO_Check_File {		
magic = 'RPCH'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RHIV' в формате ASCII.
non_revoked_certificate_flag: byte	Значение равно 0x00 в том случае, если аннулирован **сертификат** ID **системы управления платформой**, идентифицируемый по запросу; в противном случае значение равно 0x01

Таблица 7.7.3.4-2 – Синтаксис файла проверки клиента системы управления платформой

Синтаксис	Количество битов	Мнемоника
ECI_PO_Client_Check_File {		
magic = 'RPCC'	32	uimsbf
non_revoked_certificate_flag	8	uimsbf
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RHIV' в формате ASCII
non_revoked_certificate_flag: byte	Значение равно 0x00 в том случае, если образ клиента, связанный с полем cosignature_id запроса, был аннулирован в соответствии с последним списком аннулирования клиентов системы управления платформой; в противном случае значение равно 0x01

7.7.3.5 Запросы веб-интерфейса API по клиентам

Сервер **оператора** должен поддерживать следующие запросы относительно клиентов, требуемые его идентификатором **системы управления платформой**:

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

- Это выражение возвращает файл версии клиента (см. таблицу 7.7.3.5-1), содержащий последнюю версию **образа клиента ECI** для клиента, идентифицируемого по показателям **vendor_id**, **client_type**. Сервер должен, как минимум, поддерживать клиенты, используемые для обслуживания его собственной **системы управления платформой**:

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

- Это выражение возвращает файл версии идентификационных данных клиента (см. таблицу 7.7.3.5-2), содержащий последнюю версию идентификационных данных **клиента ECI** для клиента, идентифицируемого по показателям **vendor_id**, **client_type**. Сервер должен, как минимум, поддерживать клиентов, используемых для обслуживания его собственной **системы управления платформой**:

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

- Это выражение возвращает последний файл **образа клиента ECI** для клиента, идентифицируемого по показателям <vendor_id, client_type, client_version_major>. Для **образа** типа image_target_id идентификатор ECI_Image_Target_Id предоставляется как параметр запроса. Сервер должен, как минимум, поддерживать **поставщиков клиентов ECI**, используемых для обслуживания его собственной **системы управления платформой**. **Хосты ECI** используют этот API для попыток получения последних данных аннулирования всех сохраняемых **клиентов ECI**:

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

- Это выражение возвращает последний файл данных аннулирования **клиентов ECI** для клиента, опознаваемого по идентификатору **vendor_id**. Сервер должен, как минимум, поддерживать клиентов, используемых для обслуживания его собственной **системы управления платформой**.

Таблица 7.7.3.5-1 – Синтаксис файла версии клиента

Синтаксис	Количество битов	Мнемоника
ECI_Client_Version_File {		
magic = 'RCVE'	32	uimsbf
client_version	16	uimsbf
emergency_download_descriptor		
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RCVE' в формате ASCII
client_version: integer	Последняя версия клиента, тип которого определяется в запросе
emergency_download_descriptor	Дескриптор **ECI_client_emergency_download_descriptor,** в котором **хост ECI** предусматривает, что маркер platform_operation_tag должен соответствовать системе управления платформой поставщика интерфейса web-api клиента, а маркер client_tag должен соответствовать образу клиента, как заявляется в параметрах веб-интерфейса API

Таблица 7.7.3.5-2 – Синтаксис файла версии идентификационных данных клиента

Синтаксис	Количество битов	Мнемоника
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
}		

Семантика

magic: byte[4]	8-битовое представление строки 'RCCV' в формате ASCII
root_version: integer	Версия корневого сертификата (как определяется в таблице 5.3-1) последних идентификационных данных **клиента ECI**
vendor_rl_version: integer	Номер версии списка аннулирования поставщика систем безопасности для последних идентификационных данных **клиента ECI**
eci_vendor_id: ECI_Vendor_Id	Идентификатор ECI_Vendor_Id (как определяется в таблице 7.6.1-2 последних идентификационных данных **клиента ECI**
client_rl_version: integer	Номер версии списка аннулирования клиентов для последних идентификационных данных **клиента ECI**
eci_client_id: ECI_Client_Series-Id	Идентификатор ECI_Client_Series_Id (как определяется в таблице 7.6.1-2) последних идентификационных данных **клиента ECI**

7.7.3.6 Запросы веб-интерфейса API по AS_setup

В том случае, если **оператор** поддерживает онлайн-регистрацию **клиентов ECI** в шифрованном режиме, должен поддерживаться следующий запрос:

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

Этот запрос возвращает файл as_setup для определенного клиента (<vendor_id,eci_client_id>) и оборудования **CPE**, определяемого по идентификатору ECI_Image_Target_Id target_id. Идентификатор eci_client_id может быть типа ECI_Client_Id или ECI_Client_Series_Id. Nonce – это значение одноразового кода, определяемое протоколом дешифрования **образа клиента ECI**. Подробная информация приводится в пункте 7.8.4.2.

7.8 Установка клиента ECI системой управления платформой

7.8.1 Сфера применения и профилирование

Система управления платформой может выбирать функции безопасности при установках **клиента ECI** и оповещать об этом при помощи флага image_encrypted_flag и онлайн-флага в файле **образа клиента ECI** (см. таблицу 7.6.1-2):

- "режим установки **клиента ECI** с незашифрованным файлом **образа клиента ECI**", в котором загружается последняя версия **клиента ECI**, указанная в оповещении и определяемая в пункте 7.2, и выполняется инициация **клиента ECI**;
- "режим установки **клиента ECI** с зашифрованным файлом **образа клиента ECI**", который в дополнение к первому режиму разрешает **системе управления платформой** зашифровать **образ клиента ECI** и провести аутентификацию, как указывается в [ITU-T J.1014]. Дешифрование **клиента ECI** определяется **хостом ECI** и включает в себя проверку версии **хоста ECI.** Таким образом, отсутствие разрешения на дешифрование неизвестных или поврежденных **хостов ECI** гарантирует конфиденциальность **клиента ECI** после дешифрования. В случае если оборудование **CPE** не подключено к онлайн-сети, требуется идентификатор ECI_Image_Target_Id. В этом случае идентификатор ECI_Image_Target_Id должен быть вручную отправлен головному узлу системы безопасности.

Протокол инициализации обеих версий **клиента ECI** определяется в оставшейся части настоящего раздела.

Системы управления платформой, работающие с онлайн-устройствами **CPE** в режиме установки с шифрованием, могут обеспечить принудительное использование последнего **клиента ECI** при помощи одноразового кода в протоколе дешифрования, генерируемого системой AS, с сервером **системы управления платформой** для **клиента ECI** (см. пункт 7.7.3.6).

Правила профилирования:

- в случае если **система управления платформой** предлагает онлайн-регистрацию (сигнализация определяется в пункте 7.2) и оборудование **CPE** получает доступ к онлайн-услугам, оборудование **CPE** использует протокол онлайн-регистрации;
- устройства **CPE**, способные принимать радиовещательные передачи, должны иметь возможность использовать радиовещательный протокол регистрации. В режиме радиовещания необходима регистрация оборудования **CPE** при начальной регистрации **системы управления платформой**;
- системы управления платформой, работающие с радиовещательными сетями, которые поддерживают устройства **CPE** без одновременного подключения к онлайн-сети, должны поддерживать радиовещательный режим регистрации. Подробная информация для пользователя, вводящего регистрационную информацию для оборудования **CPE,** должна соответствовать действующим правилам форматирования.

7.8.2 Режим установки клиента ECI с использованием незашифрованного файла образа клиента ECI

При запуске инициализации **клиента ECI хост ECI р**езервирует **AS-сегмент** для **системы управления платформой**, сбрасывает **AS-сегмент** и загружает открытый ключ **системы управления платформой** в **AS-сегмент**, как определяется в [ITU-T J.1014].

При необходимости **хост ECI** загружает **клиента ECI,** сохраняет его в энергонезависимой оперативной памяти (NV RAV) для последующего извлечения и запускает его. Кроме того, **клиент ECI** дает **пользователю** инструкции в процессе установки. В случае если оборудование **CPE** не имеет подключения к онлайн-сети для защищенной регистрации радиовещательной системы, в процессе установки может быть предусмотрена ручная отправка **пользователем** идентификатора target_id из значения ECI_Image_Target_Id оборудования **CPE** на головной узел.

При любой последующей перезагрузке **хост ECI** будет выполнять повторную инициализацию **клиента ECI**.

7.8.3 Режим установки клиента ECI с использованием зашифрованного файла образа клиента ECI

В этом режиме работы используется шифрованная загрузка **образа клиента ECI** с помощью ключа, выбранного **оператором**. Ключ, выбранный **оператором**, зашифровывается и содержится в структуре **as_setup**.

При запуске инициализации **клиента ECI хост ECI** резервирует **AS-сегмент** для **системы управления платформой**, сбрасывает AS-сегмент и загружает открытый ключ **системы управления платформой** в **AS-сегмент**.

- **Хост ECI** различает два режима извлечения **as_setup Режим регистрации** – этот режим вводится, если **клиент ECI** инициируется впервые либо при изменении POPK или версии клиента ECI, а также если клиент работает в режиме повторной онлайн-регистрации, используя уникальный одноразовый код для каждой повторной регистрации. Структура **as_setup** для **CPE** извлекается из сети **системы управления платформой**.
- **Зарегистрированный режим** – предыдущая структура **as_setup** извлекается из энергонезависимой памяти. Если предстоит изменение версии **клиента ECI** или **хоста ECI**, **клиент ECI** должен предупредить **пользователя** о необходимости инициализации или разблокировки такой загрузки (настройки загрузки по умолчанию должны, как правило, выполняться автоматически в течение разумно обоснованного периода времени). Для загрузки нового **клиента ECI** также потребуется новая структура **as_setup**.

В режиме регистрации **хост ECI** выполняет следующие действия для извлечения новой структуры **as_setup**.

1) Хост **ECI** инициализирует AS-сегмент и извлекает:
   - идентификатор target_id из значения ECI_Image_Target_Id target_id оборудования **CPE**;
   - при онлайн-регистрации одноразовый код (128 битов) извлекается из **AS-сегмента** при помощи функции **getAsSlotRk** (см. [ITU-T J.1014]).

2) **Хост ECI** направляет вышеуказанную информацию для извлечения сообщения **as_setup** из **системы управления платформой**:
   - Если регистрация производится через **радиовещательный** канал, **хост ECI** представляет target_id на экране с диалоговым окном регистрации **системы управления платформой**. **Хост ECI** извлекает структуру **as_setup** из карусели установки AS (см. пункт 7.7.2).

   ПРИМЕЧАНИЕ 1. – В случае если платформа предоставляет несколько типов **клиентов ECI**, **система управления платформой** может запросить у **пользователя** дополнительную информацию, чтобы предоставить структуру **as_setup** для соответствующего типа **клиента ECI**.

   ПРИМЕЧАНИЕ 2. – **Система управления платформой** может предположить, что оборудование **CPE** загрузило последнюю версию **образа клиента ECI** и предоставить структуру **as_setup** только для этого **образа клиента ECI**.

   - При онлайн-**регистрации** оборудование **CPE** регистрирует идентификационные данные клиента, target_id оборудования **CPE** и одноразовый код, используя web-интерфейс API согласно пункту 7.3.3.

   ПРИМЕЧАНИЕ 3. – **Система управления платформой** может принять решение о применении одноразового кода для возобновления регистрации при каждой повторной инициализации **хоста ECI**.

Вслед за последовательностью получения **as_setup** в режиме регистрации или после восстановления структуры **as_setup** из энергонезависимой памяти после прохождения регистрации, х**ост ECI** инициализирует AS и предпринимает попытку загрузить зашифрованного **клиента ECI**.

1) **Загрузка** структуры **as_setup** в AS с использованием сообщения **reqAsClientImageDecrKey**. Загрузка цепочки сертификатов **клиентов ECI** в систему AS. Загрузка списка аннулирования клиентов **системы управления платформой** и совместной подписи клиента **системы управления платформой**. О следующих случаев сбоев необходимо как минимум уведомить **пользователя** в доступной форме или обработать их автоматически:
   a) устаревшая версия **хоста ECI** – **хост ECI** или его идентификационные данные должны быть обновлены.
   b) устаревшая версия **клиента ECI** – клиент **ECI** или его идентификационные данные должны быть обновлены.

2) Дешифрование образа с использованием в случае необходимости вычисленного ключа **образа** клиента или аутентификация **образа клиента ECI** с использованием подписи **клиента ECI** и совместных подписей клиента с**истемы управления платформой**.

3) Сбой в случае ошибки подтверждения.

Структура as_setup и формат as_setup_file должны соответствовать определению, приведенному в таблице 7.8.3-1.

Таблица 7.8.3-1 – Структура AS-Setup, файл и файл блоков данных

Синтаксис	Количество битов	Мнемоника
ECI_As_Setup {		
as_version	8	uimsbf
если (as_setup_version == 0x01) {		
vendor_id	20	uimsbf
если (/* регулярный образ клиента */){		
ECI_Client_id **client_id**		
}		
если (/* серия образов клиента */){		
ECI_Client_Series_Id **series_id**		
}		

Таблица 7.8.3-1 – Структура AS-Setup, файл и файл блоков данных

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
если (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
если (as_setup_version == 0x01){ для (i=0; i<n; i++) {		
ECI_As_Setup **as_setup_item**		
}		
}		
}		

Семантика

vendor_id: integer	Поставщик систем безопасности **клиента ECI**, для которого предназначена данная структура as_setup
client_id: ECI_Client_Id	Идентификатор **клиента ECI**, для которого предназначена данная структура as_setup. Предыдущее выражение if использует type-field client_id: оно должно соответствовать "типовому образу клиента"
series_id: ECI_Client_Series_Id	Идентификатор **серии клиента ECI**, для которой предназначена данная структура **as_setup**. Предыдущее выражение if использует идентификатор type-field client_id: он должен соответствовать "серии образов клиента"
target_id: ECI_Image_Target_id	Идентификатор ECI_Image_Target_Id, определяющий оборудование **CPE**, для которого предназначено данное сообщение
as_tag: integer	Маркер, обозначающий версию структуры as_setup для вышеуказанного целевого объекта. Это значение должно изменяться при каждом изменении структуры **as_setup** для данного целевого объекта, например – увеличиваться
online: bool	Если "истина" – то это сообщение требует использования одноразового кода сегмента в механизме ключа авторизации (AK); если "ложь – то одноразовый код не требуется. ПРИМЕЧАНИЕ. – Этот бит устанавливается только при наличии подключения к интернету
min_root_state: minEciRootState	Минимальное состояние корневого сертификата (минимальный номер версии, минимальный номер списка аннулирования) должно применяться для подтверждения загруженных **хоста ECI** и **клиентов ECI**. Это поле кодируется в виде последовательности байтов, как определяется в [ITU-T J.1014]
inputV: InputV	Сообщение InputV для системы AS. Это поле кодируется в виде последовательности байтов, как определяется в [ITU-T J.1014]
eKey: SymKey	Зашифрованный симметричный ключ для дешифрования образа. Это поле кодируется в виде последовательности байтов, как определяется в [ITU-T J.1014]
extension: Extension	Данные по расширению, обратно совместимые. В целях обеспечения компактности каруселей радиовещания указанные данные не должны превышать 256 байтов для радиовещательных приложений. Для этих данных никаких приложений не определяется
magic_file: byte[3]	8-битовое представление строки 'AES' в формате ASCII
as_setup_file_version: integer	Версия формата ECI_AS_Setup_File. Значения 0 и 0x2..0xff резервируются. Значение 0x01 используется для формата, определяемого в данном разделе
as_setup: ECI_As_Setup	Структура **as_setup системы управления платформой** для загрузки специального зашифрованного **клиента ECI** на конкретный **хост ECI**
magic_bucket_file: byte[3]	8-битовое представление строки 'AEB' в формате ASCII

as_setup_item: ECI_As_Setup	Структуры as_setup в этом блоке данных. Новые структуры as_setup добавляют в верхушку блока данных, так что самая ранняя структура находится на его дне. При необходимости структуры as_setup удаляют, но только со дна блока данных. Это позволяет устройствам **CPE** проводить более быструю проверку обновлений. Таким образом, после первой проверки достаточно только проверить структуры as_setup сверху вниз до тех пор, пока не встретится первая из проверенных ранее структур

Минимальная частота проверок для обновлений структуры **as_setup** должна быть такой же, как и для других данных **клиента ECI**, как определяется в пункте 7.3.1. Следует отметить, что под обновлением, как правило, подразумевается обновление **клиента ECI** и/или программного обеспечения **хоста ECI** для оборудования **CPE**; таким образом любые из указанных обновлений также должны быть загружены, что позволит завершить согласованную последовательность инициализации **клиента ECI**. Если новая согласованная последовательность недоступна, можно использовать предыдущую.

В то время как **хост ECI** предпринимает попытки завершения (вручную) регистрации нового или обновленного **клиента ECI** в режиме радиовещания, **хост ECI** должен проверять обновление карусели файла as_setup с максимально возможной частотой.

7.8.4 Транспортный протокол

7.8.4.1 Радиовещательный протокол

Радиовещательный протокол для структур **as_setup** должен соответствовать положениям пункта 7.7.2.

Количество структур as_setup, которые необходимо обновить при изменении версии **клиента ECI**, может быть весьма существенным. В целях ограничения количества новых онлайн-сообщений as_setup при изменении версии **клиента ECI** в режиме "только радиовещание", **система управления платформой** может предоставлять **нового клиента ECI** и *осуществлять* перегон новых идентификационных данных, тем самым заменяя группы **клиентов ECI** на устройствах **CPE**; а также может повторять данную операцию несколько раз для охвата максимально возможного количества устройств **CPE**, прежде чем использовать систему безопасности для принудительного использования нового **клиента ECI**.

7.8.4.2 Онлайн-протокол

Онлайн-протокол основан на простом протоколе запрос–отклик между оборудованием **CPE** и **клиентом ECI**, как определяется в пункте 7.7.3, передавая идентификатор **CPE target_id** и **одноразовый код** в составе запроса, а также возвращая файл **ECI_As_Setup_File**.

7.8.5 Представление пользователю идентификатора целевого объекта

Как **хост ECI**, так и **клиент ECI** должны быть способны предоставлять **пользователю** идентификатор target_id оборудования **CPE** по сетям радиовещания в случае отсутствия подключения к интернету. Они обеспечивают возможность генерации конкретной информации для оборудования **CPE**, требуемой для дешифрования **образа клиента ECI**, а также позволяют AS-системе **клиента ECI** генерировать сообщения InitV (транспортный протокол для этих сообщений определяется **клиентом ECI**). Кроме того, target_id может быть доступен для чтения в печатном виде на корпусе оборудования **CPE** или в сопроводительной документации. В данном разделе определяется порядок представления target_id для **пользователя**.

Идентификатор target_id – целое 64-битовое число. Target_id представляется **пользователю** в соответствии с правилами, изложенными в пункте 6.2.2, с применением 9-битовой контрольной суммы и добавлением 9-битовых подстрок вместо 5-битовых подстрок. Таким образом target_id представляется в виде последовательности шести четырехзначных чисел с цифрами от 0 до 7.

Устройствам **CPE** и **клиентам ECI** разрешено использовать индивидуальные представления в своих **пользовательских** интерфейсах (например, на основе частной схемы нумерации оборудования **CPE**); при этом они всегда должны предоставлять функции регистрации **клиента ECI** на основе вышеуказанного формата представления.

8 Аннулирование

8.1 Введение

Все участники **экосистемы ECI** и все внедряемые ими в экосистему элементы сертифицируются **доверительным органом ECI**. Сертификация позволяет обеспечить подходящую качественную основу как для функциональных возможностей и устойчивости реализаций, так и для принятия участниками необходимых мер по обновлению. Кроме того, этот процесс сертификации позволяет предотвращать хакерскую и пиратскую деятельность с использованием экосистемы **ECI**.

Интерфейс **ECI** обеспечивает функциональные возможности для выборочного исключения услуг, предоставляемых устройствам **CPE**, на основе статуса, присвоенного **доверительным органом ECI** оборудованию **CPE**, **хосту ECI**, другим **системам управления платформой** и загруженным **клиентам ECI**.

Доверительный орган ECI может аннулировать ту или иную **систему управления платформой**, если эти системы не выполняют общепринятые правила, в том числе правило о том, что **системы управления платформой**, совместно использующие оборудование **CPE**, не должны создавать помех друг другу, а также правило о недопустимости доставки пиратских услуг с использованием интерфейса **ECI**. Аналогичным образом **доверительный орган ECI** может аннулировать **клиентов ECI**, если они не выполняют общепринятые правила, в том числе правило о том, что **клиенты ECI**, совместно использующие оборудование **CPE**, не должны создавать помех друг другу, а также правило о недопустимости хакерской деятельности. Кроме того, **доверительный орган ECI** может аннулировать версии программного обеспечения **хоста ECI**, имеющие существенные недостатки, которые могут нарушить конфиденциальность **клиентов ECI** или допустить мошенничество.

Во всех вышеперечисленных случаях организации, ответственные за аннулируемый элемент, могут устранить указанные недостатки. Как правило, для этого необходимо заменить аннулируемый элемент на новый. **Поставщик систем безопасности** может заменить **клиента ECI** новой версией, **производитель CPE** может предоставить корректировки системы безопасности для **хоста ECI**, а **оператор** может усовершенствовать его операции, выполняемые в новой версии **сертификата системы управления платформой**. Все эти операции выполняются на основе сотрудничества с соблюдением лицензионных соглашений, заключенных между затронутыми участниками и **доверительным органом ECI**.

В случае если участники, использующие интерфейс **ECI**, допускают систематические нарушения соглашений с **доверительным органом ECI**, что отрицательно влияет на других участников или на **пользователей**, все внесенные этими участниками элементы могут быть аннулированы **доверительным органом ECI**.

Устройства **CPE**, которым более не принадлежит действующий **хост ECI** и для которых не планируются обновления от их **производителя CPE**, могут быть аннулированы в этой связи. Это происходит также в том случае, если системный загрузчик **CPE** взломан и разрешена загрузка несовместимого программного обеспечения **хоста ECI**.

Устройства **CPE** должны предпринимать попытки автоматической замены аннулируемой версии на обновленную (при наличии). Однако новые загрузки и **списки аннулирования** могут быть заблокированы. В этом случае **система управления платформой** может запретить оказание услуг или рендеринг контента, сохраненного локально на подобном устройстве **CPE**.

8.2 Аннулирование оборудования CPE

Интерфейс **ECI** разрешает **системам управления платформой** прекращать оказание услуг конкретным **устройствам CPE**, используя функции системы CA или DRM по предоставлению прав выбора. **Система управления платформой** может проверить последнее состояние, присвоенное устройству **CPE доверительным органом ECI**. В случае если **доверительный орган ECI** считает необходимым аннулировать оборудование **CPE**, **система управления платформой** может приостановить оказание услуг оборудованию **CPE** на основе его зарегистрированного идентификатора чипсета, используя услуги доставки системы CA или DRM.

Настоящая Рекомендация также облегчает **системам управления платформой** процедуру по прекращению оказания услуг тем устройствам **CPE**, на которых работают аннулируемые **хосты ECI**. **Система управления платформой** может использовать усовершенствованную систему безопасности для запроса минимального номера версии для **хоста ECI** в соответствии с последней версией списка аннулирования **хоста ECI**, как определяется в пункте 8.3.

Механизм аннулирования **хоста ECI** может также использоваться для аннулирования оборудования **CPE**, если это будет сочтено целесообразным. Аннулирование производится путем указания более высокой минимальной версии **хоста ECI**, чем та, которая была выпущена к текущему моменту.

8.3 Общий процесс аннулирования

В этом разделе говорится о комбинации минимальной версии **корневого сертификата** и минимальной версии списка аннулирования **корневого сертификата** как о минимальной версии **списка аннулирования**.

Конечным механизмом принудительного аннулирования для **хоста ECI** является ограничение услуг: в случае если аннулируемый элемент присутствует на **хосте ECI** несмотря на применение (предположительно устаревших) **списков аннулирования**, **система управления платформой** может принять решение о прекращении оказания услуг этому **хосту ECI**. Рассылка минимально допустимой версии списка аннулирования, требуемого **системой управления платформой**, защищается **системой AS**: любые манипуляции автоматически приводят к ограничению услуг. Таким образом **система управления платформой** может принудительно проверять версию идентификационных данных, используемых при установке **хоста ECI** и всех прочих **систем управления платформой** и **клиентов ECI**.

Система управления платформой предоставляет услугу загрузки **списка аннулирования** для любого из вышеуказанных элементов (**хосты ECI**, **клиенты ECI** и **системы управления платформой**). Это обеспечивает доступность последних версий списков аннулирования для всех **клиентов ECI** и систем управления платформой, загруженных на **хост ECI**.

Инициирование **AS-системы** [ITU-T J.1014] позволяет **хосту ECI** указывать данную минимальную ожидаемую версию **списка аннулирования** для всех элементов. Она применяется для ретроспективного подтверждения версии списка аннулирования, используемой **хостом ECI**. **Хост ECI** использует минимальное значение **корневого списка аннулирования** для **элементов клиента ECI**, которые он намерен загрузить, и загружаемого им **образа хоста ECI**.

ПРИМЕЧАНИЕ. – Предполагается, что **хост ECI** не загружает элементы, которые могут быть аннулированы, и уведомляет об этом **пользователя**.

Для предотвращения неправомерного ограничения услуг требуется наличие последних идентификационных данных (и, при необходимости, последних версий) для всех загружаемых элементов, которые должны быть доступны в **хосте ECI**. **Хост ECI** предоставляет нижеследующие функциональные возможности для обеспечения доступности последних идентификационных данных и (при необходимости) элементов в целях предотвращения неправомерного ограничения услуг, что позволит исключить ситуации, когда **клиенты ECI** не могут нормально функционировать из-за угроз безопасности, вызванных наличием аннулированных **сертификатов хостов ECI**, **систем управления платформой** или **клиентов ECI**.

- **Хост ECI** хранит последнюю **цепочку списков аннулирования доверительного органа ECI** для каждого элемента, который подвергается проверке в текущей конфигурации **хоста ECI**, **системы управления платформой** и **клиента ECI** с использованием услуг загрузки идентификационных данных и **списков аннулирования**, предоставляемых **производителем CPE** и **системой управления платформой** для **клиентов ECI**.

- Во всех соответствующих режимах работы оборудования **CPE** такая загрузка включена по умолчанию.
- Оборудование **CPE** не должно иметь режима работы, в котором постоянно предотвращается загрузка, кроме случаев, когда отключено питание или запрещен доступ к сети загрузки (что не связано с состоянием или режимом работы оборудования **CPE**).
- **Пользователь** должен иметь возможность беспрепятственного восстановления настроек по умолчанию, связанных с загрузкой и аннулированием по умолчанию **клиентов ECI** и **систем управления платформой**.

Настоящая Рекомендация позволяет **пользователям** корректировать режимы работы хоста по умолчанию в целях аннулирования элементов, вызывающих ограничение услуг. Если **пользователи** задействуют такую возможность (например, сохраняют старую работающую версию клиента), оказание услуг на уровне современных требований может быть затруднено.

8.4 Аннулирование хоста ECI на основе списков аннулирования

Хост ECI оборудования **CPE**, которое не обслуживается должным образом, может быть аннулирован. **Производители оборудования CPE** должны предоставлять обновленные идентификационные данные, включая последний применимый **список аннулирования ECI**. Кроме того, **система управления платформой**, которая намерена управлять **клиентом ECI** на **хосте ECI**, может предоставлять услугу загрузки **списка аннулирования**, соответствующего идентификационным данным **хоста ECI**, а также услугу загрузки для выбранных **хостов ECI**. **Хост ECI** применяет **списки аннулирования** к идентификационным данным **хоста ECI** (**корневой сертификат** и **сертификат производителя**) в соответствии с общими правилами обработки **списка аннулирования**, как определяется в [ITU-T J.1014].

Формат файла данных аннулирования **хоста ECI** определяется в пункте 5.3.

8.5 Аннулирование системы управления платформой ECI

Система управления платформой, которая намерена управлять **клиентом ECI** на **хосте ECI**, может предоставлять услугу загрузки для **списка аннулирования**, соответствующего идентификационным данным другой **системы управления платформой**. **Хост ECI** применяет **списки аннулирования** к идентификационным данным всех установленных **систем управления платформой** в соответствии с общими правилами обработки **списков аннулирования**, как определяется в [ITU-T J.1014].

Формат файла данных аннулирования **системы управления платформой ECI** определяется в пункте 7.6.3.

8.6 Аннулирование клиентов ECI

Система управления платформой, которая намерена управлять **клиентом ECI** на **хосте ECI**, может предоставлять услугу загрузки для **списка аннулирования**, соответствующего другим **клиентам ECI**. **Хост ECI** применяет **списки аннулирования** к идентификационным данным всех установленных **клиентов ECI** в соответствии с общими правилами обработки **списка аннулирования**, как определяется в [ITU-T J.1014].

Формат файла аннулирования **клиентов ECI** определяется в пункте 7.6.3.

9 Интерфейсы клиента ECI

9.1 Введение

9.1.1 Архитектура интерфейсов клиента ECI

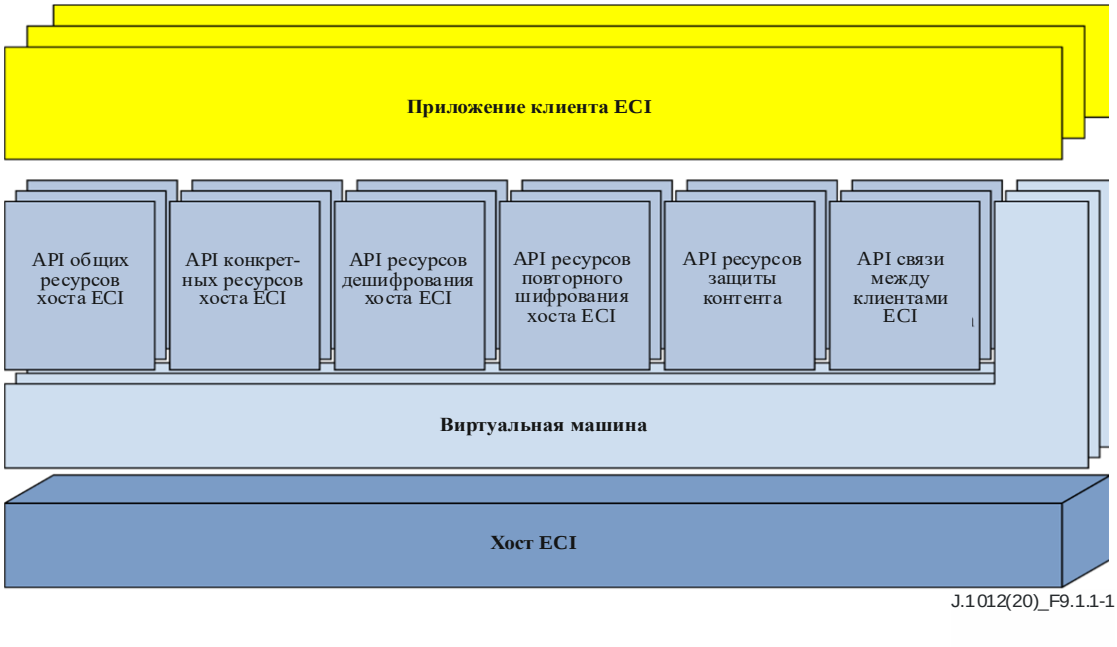


Рисунок 9.1.1-1 – Структура интерфейсов API, определяемая в разделе 9

На рисунке 9.1.1-1 дается общее представление о структуре интерфейсов API системы **ECI**. Данная структура включает в себя шесть блоков интерфейсов API, которые могут использоваться **клиентом ECI**. Эти блоки интерфейсов API определяются в пунктах 9.4–9.9. В таблице 9.1.1-1 приведен перечень интерфейсов API, определяемых в разделе 9 настоящей Рекомендации; см. также [b-ETSI GS ECI 002].

Таблица 9.1.1-1 – Список интерфейсов API, определяемых в настоящей Рекомендации

Номер пункта	Категория API	Описание
9.4	Интерфейсы API для общих ресурсов **хоста ECI**	Интерфейсы API, поддерживающие общие функциональные возможности **клиента ECI**
9.5	Интерфейсы API для конкретных ресурсов **хоста ECI**	Интерфейсы API, поддерживающие конкретные функциональные возможности **клиента ECI**
9.6	Интерфейсы API для доступа к ресурсам дешифрования **хоста ECI**	Интерфейсы API, позволяющие **клиенту ECI** использовать ресурсы дешифрования **хоста ECI**
9.7	Интерфейсы API для доступа к ресурсам повторного шифрования **хоста ECI**	Интерфейсы API, позволяющие **клиенту ECI** использовать ресурсы повторного шифрования **хоста ECI**
9.8	Интерфейсы API для ресурсов, связанных со свойствами контента	Интерфейсы API, поддерживающие функциональные возможности **клиента ECI**, связанные с защитой контента
9.9	Интерфейсы API для связи между **клиентами ECI**	Интерфейсы API, поддерживающие прямую связь между **клиентами ECI**

9.1.2 Указатель медиаданных

Указатель медиаданных (Media Handle) – это идентификатор объекта в среде хоста, который обеспечивает контекст для всех интерфейсов **хоста ECI**, предоставленных **клиенту ECI** в рамках управления процессом дешифрования элемента контента. Кроме того, **указатель медиаданных** позволяет **клиенту ECI** определять требуемые ему данные, содержащиеся в контейнере контента, что дает возможность дескремблировать контент. В случае рассылки контента по сетям радиовещания указатель медиаданных контролирует выбор программы для декодирования и выбор потока из сети рассылки (функция настройки). **Клиент ECI** также может отправлять запрос **указателю медиаданных**, имеющему доступ к устройству настройки, для получения доступа к данным, необходимым для работы **клиента ECI**, из сетевых потоков, недоступных для приложения или хоста, в целях приобретения контента. Для рассылки файлов и данных на основе OTT-потоков **указатель**

медиаданных предоставляет **клиенту ECI** доступ к данным системы безопасности в файле или потоке, не определенном в стандартном месте расположения.

Дескремблирование сеанса передачи медиаданных осуществляется непосредственно под управлением **клиента ECI**. Синхронизация применения контрольных слов в транспортном потоке основана на скремблировании сигналов управления в транспортном потоке. Синхронизация контрольных слов (в этом контексте обычно называемых ключами) и файла ISOBMFF CENC [ISO/IEC 23001-7] основана на идентификаторах CENC KeyID.

Сеансы, работающие с **указателем медиаданных**, перечисляются в таблице 9.1.2-1.

Таблица 9.1.2-1 – Типы указателей медиаданных

Название	Значение	Описание
MhDvbTs	0x01	Транспортный поток должен соответствовать [ISO/IEC 13818-1-1]
MhIsobmffCenc	0x10	Файл ISO BMFF должен соответствовать [ISO/IEC 23001-9] и [ISO/IEC 14496-12]
RFU	Прочее	Зарезервировано для использования в будущем

9.2 Интерфейс виртуальной машины ECI

9.2.1 Принципы

Для каждого **клиента ECI** создается отдельная копия виртуальной машины. Загрузка данных и инструкций для **клиента ECI** в виртуальную машину (VM) определяются в разделе 7.

Режим работы виртуальной машины определяется в [ITU-T J.1013]; см. также [b-ETSI GS ECI 001-4].

Клиент ECI взаимодействует с внешней средой при помощи интерфейса сообщений, как определяется в пункте 9.2.3.

9.2.2 Инструкции и данные (статические ресурсы)

Виртуальная машина выполняет инструкции, предоставляемые ей **загрузчиком клиента ECI** как часть сегмента(ов) кода **образа клиента ECI**.

Виртуальная машина обеспечивает отсутствие самопроизвольного изменения инструкций. Любой код, который легко приводит к нежелательному и/или доступному для воздействия режиму работы **клиента ECI** (например, интерпретаторы), считается неподходящим и должен быть включен в процесс сертификации **клиентов ECI**.

Максимальный размер кода и пространство статических данных, требуемые **клиентом ECI**, предлагаются в [b-ITU-T J Suppl. 7].

9.2.3 Взаимодействие с хостом ECI

Все виды взаимодействия между **клиентом ECI** и **хостом ECI** определяются на основе модели сообщения в данном разделе. **Клиент ECI** и **хост ECI** не используют совместно данные, за исключением:

- данных, содержащихся в сообщениях;
- любых данных, хранящихся в энергонезависимой памяти **хоста ECI** и связанных с **клиентом ECI**; или
- любых данных в сообщениях, передаваемых по каналам связи другим **клиентам ECI** или от них.

Следует отметить, что эти данные также передаются при помощи сообщений.

В основе модели сообщения лежат три разных типа обмена сообщениями между **клиентом ECI** и **хостом ECI**.

1) **Синхронный клиент** инициирует обмен следующим образом: **клиент ECI** вызывает **функцию хоста ECI**, которая откликается через очень короткий период времени. В то время как **хост ECI** обрабатывает сообщение и передает сообщение с кодом возврата, управляющий поток **клиента ECI** блокируется.

2) **Асинхронный клиент** инициирует обмен следующим образом: **клиент ECI** направляет **хосту ECI** сообщение с **запросом клиента**, которое ставится в очередь и обрабатывается в установленном порядке **хостом ECI**. Асинхронный вызов незамедлительно передает код **возврата**, содержащий только основной результат (идентификатор сообщения или ошибку). Затем **хост ECI** возвращает сообщение-**отклик хоста**, содержащее статус и результаты действий **хоста ECI**, инициированного **клиентом ECI**.

3) **Асинхронный** хост инициирует обмен следующим образом: **хост ECI** направляет **клиенту ECI** сообщение, которое ставится в очередь и обрабатывается в установленном порядке **клиентом ECI**. Асинхронный вызов незамедлительно передает код возврата, содержащий только основной (стандартный) результат. Тип и формат этого сообщения в виде, представленном в **хосте ECI**, выходят за рамки настоящей Рекомендации, так как относятся к внутренним аспектам **хоста ECI**.

Следует иметь в виду, что определяется только представление сообщения для клиента ECI. Затем клиент ECI передает сообщение – отклик, содержащее статус и результаты функционирования клиента ECI, инициированного хостом ECI.

На рисунке 9.2.3-1 показаны различные виды обмена сообщениями между **хостом ECI** и **клиентом ECI**.

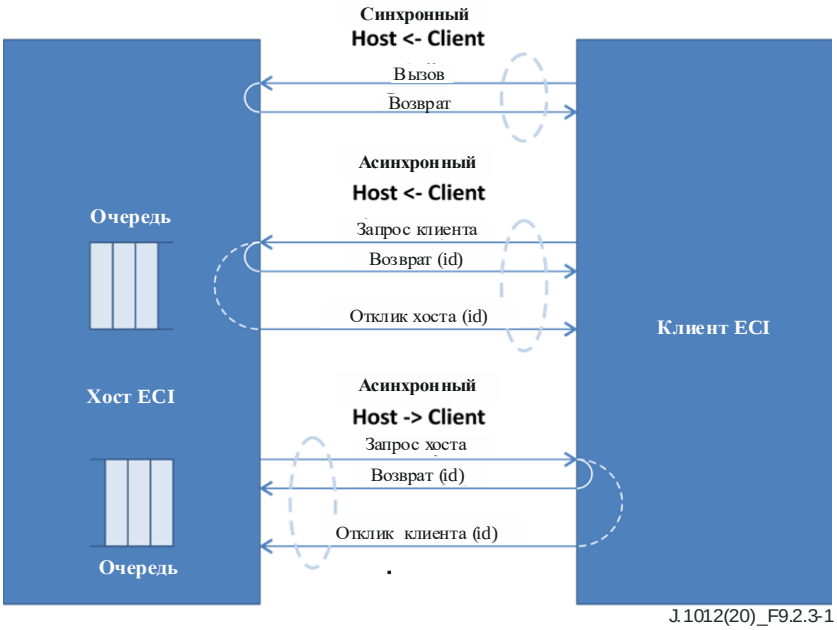


Рисунок 9.2.3-1 – Виды обмена сообщениями между клиентом и хостом

Клиент ECI должен обеспечить надлежащую защиту полезных данных, таких как контрольные слова и свойства контента. Кроме того, этот интерфейс не разрабатывался и не предназначался для обмена контентом.

Клиент ECI реализует отклики на **запросы хоста ECI**, которые он поддерживает в соответствии с определениями интерфейса API, приведенными в разделе 9, при помощи идентификатора **запросов**, содержащегося в **отклике**.

Хост ECI реализует отклики на **запросы клиента ECI**, которые он поддерживает в соответствии с определениями интерфейса API, приведенными в разделе 9, при помощи идентификатора **запросов**, содержащегося в **отклике**.

Асинхронный **запрос** может в ряде случаев указывать на то, что **отклик** не требуется. Например, при массовом перемещении элементов данных инициатор требует только **отклик** на последний **запрос**, подразумевая, что все промежуточные элементы данных обработаны корректно.

Все асинхронные **запросы** и **отклики хоста ECI** ставятся в очередь в порядке их поступления.

9.2.4 Динамические ресурсы, предоставленные клиенту ECI

Технические параметры минимально требуемых динамических ресурсов **клиента ECI** предлагаются в [b-ITU-T J Suppl. 7]. Спецификация охватывает следующие элементы: управляющие потоки, стековое пространство, объем динамической области, время выполнения, энергонезависимое хранилище и передачу сообщений между клиентами.

9.2.5 Управление версиями API

Интерфейсы API, определяемые в настоящей Рекомендации, могут иметь несколько версий, например, в целях предоставления расширенного набора функций, который заменяет предыдущий набор функций, или для устранения недостатка спецификаций. При инициировании **клиентов ECI** и их **хоста ECI** необходимо установить, какие интерфейсы API поддерживаются их ответным элементом, и выбрать, какая из версий каждого из доступных API ответного элемента будет использоваться на протяжении оставшегося жизненного цикла **клиента ECI**. **Клиенты ECI** не могут использовать другие интерфейсы API кроме API-интерфейсов обнаружения на этапе инициирования, поскольку версии сообщений (то есть их готовность, длина и синтаксис) не определяются, пока не завершится процесс обнаружения.

Версии API обладают замкнутой семантикой, то есть обмен сообщениями между **клиентом ECI** и **хостом ECI** через версию API не зависит ни от поддержки других версий этого API в **хосте ECI**, ни от обмена данными между **хостом ECI** и другими **клиентами ECI** с использованием других версий этого API.

ПРИМЕЧАНИЕ 1. – По практическим соображениям текст в разделах, в которых определяются новые версии API, может содержать ссылки на текст, определяющий более ранние версии API в настоящей Рекомендации.

Интерфейсы API могут быть обязательными, необязательными или условными (то есть обязательными при определенном условии). Примером зависимости от условий может являться то, что применяемый в PVR интерфейс API должен поддерживаться в PVR-совместимом оборудовании **CPE**. В будущих версиях настоящей Рекомендации могут определяться профили интерфейсов API, которые должны поддерживаться **хостами ECI** и **клиентами ECI**, ссылающимися на название профиля и номер версии спецификации.

В целях соответствия настоящей Рекомендации и обеспечения обратной совместимости **хост ECI** или **клиент ECI**, поддерживающие API, должны поддерживать все версии этого API (включая последние), если только более ранние версии явно не определяются как устаревшие в настоящей Рекомендации (как и в будущих ее вариантах) или если явно не указано иное.

ПРИМЕЧАНИЕ 2. – Разработка будущей версии настоящей Рекомендации не подразумевает, что действующие или обновленные **клиенты ECI** или **хосты ECI** должны соответствовать требованиям. Принципы обновления полей **хостов ECI** и **клиентов ECI** до новых версий спецификаций или правил, которые устанавливают новые версии спецификаций в качестве обязательных и применяются к новым **хостам ECI** и **клиентам ECI**, выходят за рамки настоящей Рекомендации.

Клиенты ECI выбирают максимальный номер версии из доступных интерфейсов API в **хостах ECI**, который они способны обработать, и наоборот, **хосты ECI** выбирают максимальный доступный номер версии интерфейса API в **клиентах ECI**, который они способны обработать. Таким образом возникает тенденция использования наиболее проработанных версий интерфейсов API, что позволяет избежать проблем с устаревшими (более ранними) версиями API.

Учитывая более длительный жизненный цикл **хостов ECI** и относительную простоту обновления **клиентов ECI**, **клиенты ECI** должны быть способны поддерживать более ранние версии интерфейса API для **хоста ECI**, что отражает базовую ситуацию (данная ситуация может быть предметом дополнительных соглашений, выходящих за рамки сферы применения настоящей Рекомендации). И наоборот, новые **хосты ECI** должны поддерживать более ранние версии **клиентов ECI**, отражающие развертывание **клиента ECI** (которое может быть предметом дополнительных соглашений, выходящих за рамки сферы применения настоящей Рекомендации).

API-интерфейс обнаружения **хост ECI – клиент ECI** определяется в пункте 9.4.2.

9.2.6 Контроль функции реагирования

Хост ECI вводит в действие ряд основных функций автоматического перезапуска **клиента ECI** в целях обеспечения дополнительной устойчивости общего набора функций оборудования **CPE**. **Хост ECI** обнаруживает состояния критических ошибок в **клиенте ECI** и автоматически повторно инициирует **клиента ECI** в случае таких событий. Перед повторным инициированием будут разблокированы все ресурсы, используемые **клиентом ECI**, включая **указатели медиаданных**, сеансы интерфейса mmi, файлы, IP-соединения и т. д.

Определяются следующие состояния ошибок.

- **Хост ECI** контролирует выполнение любой нелегальной инструкции кодом **клиента ECI**, наподобие неопределенного кода команд, рассмотрения недопустимых данных или рассмотрения несуществующего кода, переполнения и незаполнения стека регистра.
- **Хост ECI** использует время ожидания при приеме нового сообщения **клиентом ECI**. Предлагаемая арактеристика для этого параметра приводится в [b-ITU-T J Suppl. 7].

В случае неоднократного повторного инициирования **хост ECI** может использовать политику с возможным применением **пользовательских** настроек или ввода данных **пользователем** для декодирования и исключения повторных ошибок **клиента ECI** на более постоянной основе.

ПРИМЕЧАНИЕ. – Любое выполнение системного вызова sys_exit (см. [ITU-T J.1013]) **клиентом ECI** будет трактоваться как обычное завершение работы **клиента ECI**. Как правило, это подразумевает, что **клиент ECI** может быть удален или заменен более поздней версией. При наступлении подобного события **хост ECI** не удаляет автоматически **клиент ECI**, а ожидает вызова соответствующей процедуры замены или удаления через другие политики управления для **клиентов ECI**.

9.3 Механизм для интерфейсов API клиента ECI

9.3.1 Синтаксис асинхронных сообщений

Все структуры сообщений определяются в соответствии с их появлением в виртуальной машине **ECI**. Структура буфера для всех асинхронных сообщений в соответствии с их появлением на карте распределения памяти виртуальной машины представлена в таблице 9.3-1. Следует отметить, что все буферы сообщений выровнены по 32-битовой границе.

Таблица 9.3-1 – Синтаксис асинхронных сообщений

Синтаксис C-style	Количество битов
struct messageBuffer {	
uint32 **msgTag**;	32
uint16 **msgId**	16
uint16 **payloadLen;**	16
uint32 **payload[];**	n*32
} MessageBuffer;	

msgTag

В этом поле представлены следующие значения:

- биты 0–15: **msgApiTag**. Идентификация API для сообщения (определение приведено в приложении C);
- биты 16–23: **msgCallTag**. Идентификация вызова API, интерпретируемая устройством приема в контексте значения **msgTag** и согласованной версии API;
- биты 24–31: **msgFlags**. Дополнительные флаги для оценки сообщения. Применяются следующие определения:
  - бит 24: **msgNoResFlag**. Для сообщений **запроса** и активации: если значение 0b1 – **отклик** или ответ не требуется; если значение 0b0 – требуется **отклик** или ответ. Этот бит не имеет значения в сообщениях, содержащих ответы и отклики;
  - биты 25–31 резервируются для использования в будущем. Инициатор сообщения задает значение этих битов, равное 0b0.

Маркер сообщения должен быть идентичным **откликам** на сообщения-**запросы** и ответам на сообщения активации.

msgId

- Значение идентификатора сообщения, присвоенное **хостом ECI**. Для сообщения-отклика это значение должно соответствовать значению исходного сообщения-запроса. **Клиент ECI**, отправляющий запрос, может оставить это поле неинициированным (соответствующее значение будет присвоено **хостом ECI** и возвращено в качестве результирующего значения системного вызова SYS_PUTMSG).

payloadLen

- В поле длины полезной нагрузки представлен размер буфера полезной нагрузки в байтах. Фактически выделенный размер поля полезной нагрузки должен быть равен этому значению, округленному до следующего числа, кратного 4, или большего. При интерпретации поля **payload** полученного сообщения **хосты ECI** проверяют, что данные не выходят за пределы **payloadLen**; в противном случае возвращается сообщение об ошибке. **Клиенты ECI** могут допускать, что **хосты ECI** представляют буферы сообщений, надлежащим образом рассчитанные по размеру.

поле payload

- Поле payload используется для передачи параметров сообщения. Структура полезной нагрузки определяется с помощью синтаксиса c-syntax для подписи вызова функции, используемой согласно конкретным правилам отображения, определяемым в пункте 9.3.2.3.

9.3.2 Правила для определения структуры асинхронных сообщений

9.3.2.1 Синтаксис определений сообщений

Асинхронные сообщения определяются с помощью декларации подписи функции c-style. Эта нотация соответствует структуре сообщений согласно правилам, определяемым в настоящем разделе. Ниже приведен пример декларации подписи функции:

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

9.3.2.2 Основные типы параметров сообщений

В синтаксисе используются основные типы для определений параметров, как указывается в таблице 9.3.2.2-1.

Таблица 9.3.2.2-1 – Основные типы определений параметров сообщений

Основные типы	Представление
uint8, uchar, byte:	8-битовое целое без знака
int8, char, bool:	8-битовое целое со знаком
uint16, ushort:	16-битовое целое без знака
int16, short:	16-битовое целое со знаком
uint32, uint:	32-битовое целое без знака
int32, int:	32-битовое целое со знаком
uint64, ulong:	64-битовое целое без знака
int64, long:	64-битовое целое со знаком
char *, … ,long * (client memory)	32-битовое; допускается только в синхронных сообщениях

Для параметров булева типа используются символьные значения **истина** (True) и **ложь** (False). В соответствии с определением c-language **False** представляется значением 0x00, **True** представляется любым значением, отличным от 0x00.

9.3.2.3 Соответствие полезной нагрузки сообщения параметру сообщения

Поле **payload** содержит все параметры для сообщения. Параметр идентификатора сообщения **msgId** и параметры результата **msgResult** относятся к неявным в том смысле, что они не показаны в декларативном описании синтаксиса подписи функции. Их наличие косвенно определяется типом сообщения.

Хост ECI связывает значение **msgId** с сообщениями запросов **хоста ECI** и **клиента ECI** для того, чтобы связать **запрос** с соответствующим ответом. Тип msgId имеет значение uint32. **Хост ECI**

отвечает за управление значениями msgId. Значения msgId не должны перевыпускаться до тех пор, пока не будет передано сообщение с **откликом**.

Отклик должен содержать параметр **msgResult** типа int32.

Эти неявные параметры являются первыми в поле полезной нагрузки буфера сообщений. Последовательность параметров поля полезной нагрузки для каждого типа сообщений с точки зрения **клиента ECI** представлена в таблице 9.3.2.3-1 (соответствующие данные с точки зрения **хоста ECI** выходят за рамки сферы применения **ECI**).

Таблица 9.3.2.3-1 – Типы сообщений и "скрытые" параметры (с точки зрения клиента)

Тип сообщения	Неявные параметры	Поле payload
Запрос клиента, C→H	*None*	p_1, .. , p_n
Отклик хоста, H→C	msgId, результат	msgId, результат, p_1, .. , p_n
Запрос хоста, H→C	msgId	msgId, p_1, .. , p_n
Отклик клиента, C→H	msgId, результат	msgId, результат, p_1, .. , p_n

Следующие правила должны применяться при преобразовании параметров (будь то структуры, байтовые и короткие последовательности и т. д.) в структуру полезной нагрузки буфера сообщений в пространстве памяти **клиента ECI**.

- Параметры отображаются в памяти (первым следует самый меньший адрес) за исключением полей данных последовательностей переменной длины.
- Любой тип 8- или 16-битовых данных расширяется до 32 битов с помощью расширения, соответствующего типу данных (со знаком или без знака).
- Структуры (кроме полей битов). Все поля должны отображаться в том порядке, в котором они определяются: первое поле по самому меньшему адресу, выровненное по размеру поля (для 16- и 32-битовых объектов), за ним следует поле заполнения, предшествующее полю большего адреса. Структура всегда заполняется до следующей 32-битовой границы. Объединенные структуры должны быть заполнены до максимального размера альтернативных структур.
- Байтовые (8-битовые), короткие (16-битовые) и int (32-битовые последовательности) должны быть включены в буфер сообщений (не в качестве указателей памяти **клиента ECI**). В последовательностях фиксированной длины используются следующие обозначения: <type>, <array_identifier>, '[' <constant> ']'. Они должны отображаться в порядке появления в списке параметров. В последовательностях переменной длины используются обозначения <type>, <array_identifier>, '[' ']'. Все массивы переменной длины должны отображаться в двух 32-битовых полях. Первое поле содержит смещение в буфере сообщений, где располагается первый элемент последовательности. Второе поле содержит длину последовательности (в байтах).
- При хранении 64-битовых объектов первым должен следовать самый старший 32-битовый объект (согласно стандартным условиям отображения 64-битовых объектов на 32-битовых машинах в формате little-endian, то есть с обратным порядком следования двоичных данных).
- Все 32- и 16-битовые объекты имеют естественное (неизвестное, определяемое базовой архитектурой процессора) представление порядка следования байтов в памяти.
- Любой знак (char*), указывающий на печатаемые символы, использует представление UTF-8 [ISO/IEC 21320] для актуальных кодовых точек, если явно не указано иное. Представление символов может занимать от 1 до 4 байтов (в зависимости от кодовой точки). Данная спецификация не определяет, какие кодовые точки должны быть печатаемыми в устройстве **CPE** (которое может по-разному эксплуатироваться в разных регионах).

ПРИМЕЧАНИЕ. – Хост ECI отвечает за интерпретацию маркера сообщения в сочетании с версией API, согласованной с клиентом ECI в процессе обнаружения. Аналогичным образом клиент ECI отвечает за интерпретацию маркера сообщения в сочетании с версией API, согласованной с хостом ECI в процессе обнаружения.

9.3.2.4 Соглашение по присвоению имен для асинхронных сообщений

Соглашения об именах функций:

Все имена функций начинаются с трехбуквенного обозначения, отражающего тип сообщения. Обозначение <name> функции начинается с заглавной буквы. Ниже описывается методика присвоения имен сообщениям по их типу:

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

ПРИМЕР 1: reqIpTcpSend().

Соглашение для обозначения парных сообщений:

Сообщения, содержащие **запросы** и **отклики**, определяются как парные, аналогично сообщениям вызова и ответа. Для обозначения таких пар сообщений используется следующая форма записи:

```
<requestMessage>  →  <responseMessage>
```

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

В этих и в других обозначениях подписи функций могут появляться без ввода параметров (для краткости).

В таблице 9.3.2.4-1 представлены некоторые примеры практического сопоставления имени сообщения с возможными функциями c-functions с использованием оформленных в виде процедур программных методов подписки на события и/или функций обратного вызова в стиле javascript, либо с использованием циклов отправки. Функция **subscr** разрешает вызов функции при приеме сообщения с маркером. Приведены два примера: первый – избирательный по идентификатору **msgId**, включающий в себя структуру **cntxt** для этой функции. Во втором примере не фильтруется **msgId** и не предоставляется структура **cntxt** при обратном вызове/отправке.

Таблица 9.3.2.4-1 – Параметры в поле payload для разных типов сообщений с параметрами p_1, .. ,p_n

Сообщение	Обозначение в виде процедуры	Подписка на события клиента с обратным вызовом	Обозначение или аннулирование обратного вызова/отправки
Req, C→H	id = reqName([tag],p_1,..p_n)		
Res, H→C	res = resName([tag],id,p_1,..p_n)	subscr(tag,id,resName,cntxt) subscr(tag,resName)	resName(cntxt,res,p_1,..p_n) resName(id,p_1,..p_n)
Req, H→C	[tag =] reqName([id],p_1,..p_n)	subscr(tag,invName)	invName(id,p_1,..p_n)
Res, C→H	resName([tag],id,res,p_1..p_n)		

9.3.3 Синхронные сообщения

Для синхронных сообщений применяется то же соглашение об обозначениях с использованием имен функций в качестве асинхронных сообщений. Параметры синхронных сообщений не должны быть упорядоченными, чтобы точно вписываться в буферы сообщений, однако должны использовать общие соглашения c-conventions для вызовов функций, а также использовать определение двоичного интерфейса приложения виртуальной машины для отображения процедуры в памяти виртуальной машины и состоянии регистра. Таким образом синхронные сообщения могут отображаться непосредственно в обычных функциях кодирования в качестве части библиотеки **клиента ECI**.

Существуют три предопределенных типа: **get** – для считывания переменной в домене **хоста ECI**, **set** – для записи переменной в домене **хоста ECI** и **call** – функция общего назначения с отрицательным кодом ошибки или возвратом неотрицательного значения функции, как показано в таблице 9.3.3-1.

Таблица 9.3.3-1 – Типы синхронных функций

Тип	Применяется для	Обозначение	Результат	Семантика
Get	Переменная хоста	getVariable((**i1**..**in**)	тип переменной	Считывание переменной, индексируемой по параметрам **i1**..**in** в домене **хоста ECI** (для данного **клиента ECI**) (см. примечание)
Set	Переменная хоста	setVariable((**i1**..**in,** value)	void	Присвоение значений переменной, индексируемой по параметрам **i1**..**in** в домене **хоста ECI** (для данного **клиента ECI**) (см. примечание)
Call	Хост	callFunc(p1..pn)	int или void	Осуществление синхронного вызова (общего назначения) функции в домене **хоста ECI.** Значение кода возврата того же типа, что и результирующее значение для асинхронных сообщений, то есть отрицательные значения свидетельствуют о возникшей ошибке. Могут встречаться функции типа void, которые не разрешают отправку сообщений об ошибках
ПРИМЕЧАНИЕ. – **Хост ECI** может быть активирован для выполнения действий в дополнение к возврату запрашиваемого объекта как следствие аннулирования функции Get.				

Примеры определения синхронного сообщения

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

Примеры использования

```
uint clock = getClock() ;    /* считать показания часов */
setPwrWakeup (1000);         /* установить таймер активации; запускает аннулирования
*/
(void) memcpy(ptr1,ptr2,100*1000)   /* копировать память клиента эффективным образом
*/
```

9.3.4 Коды ошибок, применяемые в коде возврата

Параметры кода возврата **откликов**, **ответов** и (если применимо) **вызовов** содержат одиночное 32-битовое число со знаком. Если возвращаемое значение больше или равно нулю, то выполнение кода успешно завершено. Отрицательное возвращаемое значение свидетельствует об ошибке. Ошибки могут быть общими (см. таблицу 9.3.4-1) или связанными с **запросами** (см. коды ошибок, связанных с **запросами**).

Таблица 9.3.4-1 – Коды ошибок для сообщений с кодом возврата

Наименование/постоянная	Значение	Описание
	1..MaxInt	**Запрос** выполнен успешно, значение задается в определениях сообщений
ErrReqOkNoId	0	**Запрос** выполнен успешно
ErrReqApiErr	–1	API, обозначенный маркером msgApiTag, не поддерживается
ErrReqCallErr	–2	Вызов для API, обозначенного маркером msgApiTag, не поддерживается
ReqQueueErr	–3	Проблема при постановке сообщения в очередь. Очередь буфера **ECI** переполнена
ReqResource	–4	Проблема с ресурсами при обработке **запроса** (например, нехватка памяти в связи с избыточным количеством сообщений)
RFU	–5..–15	Зарезервировано для использования в будущем (общие типы ошибок)
ReqParam<N>Err	–16..–48	Ошибка параметра **N** = -Result-15
Зарезервировано для ошибок виртуальной машины	–49..–64	Коды ошибок резервируются для ошибок виртуальной машины, как определяется в [ITU-T J.1013]
RFU	–65 .. –256	Зарезервировано для использования в будущем
Ошибка интерфейса API	–256 .. –511	Ошибка API, указанная в таблице кодов ошибок API
RFU	–512.. MinInt	Зарезервировано для использования в будущем

ПРИМЕЧАНИЕ. – Как правило, **клиент ECI** может опираться на **хост ECI** в целях поддержки конкретного профиля интерфейсов API, как определяется в пункте 9.2.5, и постановки в очередь буферов сообщений для обеспечения запаса. Следовательно, в большинстве случаев настраиваемая обработка ошибок не требуется; код ошибки, как правило, необходим только для сценариев отладки **клиента ECI**.

Коды ошибок, связанные с API, или ReqParamNErr, не могут быть возвращены как часть кода возврата, однако вместо этого ошибка такого типа должна передаваться как часть **отклика**.

9.3.5 Защищенный аутентифицированный канал

Средства для установления **защищенного аутентифицированного канала (SAC)** между **клиентом ECI** и любым другим соответствующим устройством доступны при использовании усовершенствованных защищенных интерфейсов API (см. пункт 9.5.2). В случае потребности **клиента ECI** в защищенной аутентифицированной связи с другим **клиентом ECI** или каким-либо внешним устройством требуется определить проприетарный механизм, способный использовать доступные интерфейсы API, в особенности усовершенствованные защищенные интерфейсы API.

9.3.6 Проверка сообщений хостом ECI

Во избежание состояний ошибки или неприемлемых действий вследствие неадекватных **запросов** или **откликов хосты ECI** выполняют полную проверку всех сообщений, полученных от **клиента ECI**. Выполняются следующие виды проверки:

- поддержка **msgApiTag**;
- поддержка **msgCallId** в пространстве сообщения API (в контексте версии API, установленной при обнаружении);
- проверка того, соответствуют ли ограничения полезной нагрузки и в особенности **msgLength** правилам синтаксиса сообщения и ограничивается ли буфер сообщений (для асинхронных сообщений) и системы памяти адресного пространства **клиента ECI** для чтения или записи со стороны **хоста ECI**, определенными частями адресного пространства **клиента ECI**;
- проверить, не нарушено ли какое-либо **предварительное условие** для сообщения (в том смысле, что **предварительное условие** имеет значение для целостности **запроса** или **отклика**);
- проверить, что любой указатель или память, задействованные в сообщении, являются памятью, выделенной **клиенту ECI**.

9.3.7 Обработка сообщений клиентами ECI

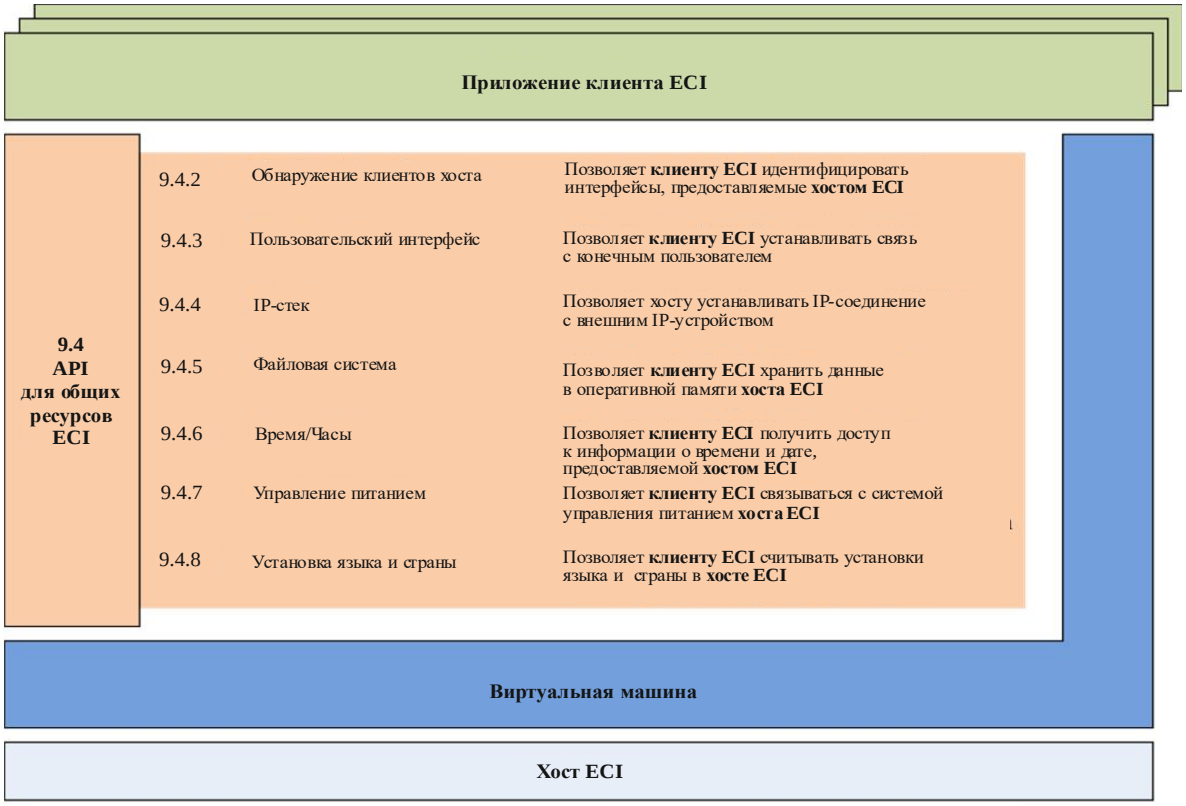
Любая память, выделенная для отправки **запроса**, может быть повторно задействована при получении кода возврата, если явно не указано иное (как правило, длинные сообщения, для которых важно избегать копирования). Аналогичным образом любая память, выделенная для отправки **отклика**, может быть повторно задействована непосредственно после отправки.

Клиенты ECI не должны полагаться на **хосты ECI** для возврата **отклика** каждого **запроса**.

Клиенты ECI могут проверить корректность синтаксиса любого **запроса** или **отклика хоста ECI**. В случае некорректно отформатированного **запроса** или **отклика клиент ECI** не обязан реагировать в целях предоставления **хосту ECI** какой-либо обратной связи.

9.4 Интерфейсы API для общих ресурсов хоста ECI

9.4.1 Список интерфейсов API, определяемых в пункте 9.4



J.1012(20)_F9.4.1-1

Рисунок 9.4.1-1 – Блок-схема интерфейсов API, определенных в пункте 9.4

Таблица 9.4.1-1 – Список интерфейсов API, определяемых в пункте 9.4

Пункт	Наименование API	Описание
9.4.2	Обнаружение интерфейса хоста	Позволяет **клиенту ECI** идентифицировать интерфейсы, предоставляемые **хостом ECI**
9.4.3	Пользовательский интерфейс	Позволяет **клиенту ECI** устанавливать связь с **пользователем**
9.4.4	Стек IP-протокола	Позволяет хосту устанавливать IP-соединение с внешним IP-устройством
9.4.5	Файловая система	Позволяет **клиенту ECI** хранить данные в оперативной памяти **хоста ECI**
9.4.6	Время/часы	Позволяет **клиенту ECI** получить доступ к информации о времени и дате, предоставляемой **хостом ECI**
9.4.7	Управление питанием	Позволяет **клиенту ECI** связываться с системой управления питанием **хоста ECI**
9.4.8	Язык и региональные настройки	Позволяет **клиенту ECI** считывать установки языка и страны в **хосте ECI**

В таблице 9.4.1-1 показаны интерфейсы API, определяемые в пункте 9.4 и на рисунке 9.4.1-1, который иллюстрирует расположение интерфейсов API, определяемых в пункте 9.4, с **архитектурой ECI**.

Обзор сообщений по представлениям, относящихся к различным интерфейсам API, приведен в таблицах для каждого интерфейса API вместе со структурой, описываемой в таблице 9.4.1-2.

Таблица 9.4.1-2 – Структура таблицы, в которой сведены функции отдельных сообщений API

Сообщение	Тип	Направление	Маркер	Описание
Имя сообщения	См. таблицу 9.4.1-3	C→H или H→C	Значение маркера	Краткое описание функции сообщения

Столбец "Тип" в таблице 9.4.1-2 указывает тип соответствующего сообщения, которое может быть как синхронным, так и асинхронным. Более подробная информация приведена в таблице 9.4.1-3. Полный список всех сообщений API, доступных для **клиента ECI**, приведен в Дополнении I.

Таблица 9.4.1-3 – Возможные значения для столбца Тип

Категория сообщения	Обозначение в столбце Тип	Комментарий
Асинхронное сообщение	A	Возможные типы сообщений: см. таблицу 9.3.2.3-1
Синхронное сообщение	A	Возможные типы сообщений: см. таблицу 9.3.3-1
	Set	
	Get	
	Call	

9.4.2 Интерфейс API для доступа к ресурсу обнаружения интерфейса хоста ECI

9.4.2.1 Введение

В настоящем разделе определяется интерфейс API, который может использоваться **клиентом ECI** для обнаружения интерфейсов и версий API, поддерживаемых **хостом ECI**, а также для выбора наиболее подходящей версии продолжительности сеанса связи **клиента ECI** с **хостом ECI**. Механизм управления версиями API позволяет выбрать нужный интерфейс API путем перебора имеющихся вариантов. Выбранная версия API будет использоваться до следующего события инициирования **клиента ECI** с **хостом ECI**.

Концепции, касающиеся доступности интерфейсов API, рассматриваются в пункте 9.2.5. Обязательные интерфейсы API определяются в разделе 10.

Клиент ECI приступает к управлению версиями сразу после инициирования: ни один API не может использоваться без (взаимно) установленной версии.

Версия API представляется 16-битовым числом. Нумерация версий API начинается с 0x0000. Как правило, новые версии нумеруются с возрастанием на 1.

Список сообщений API приводится в таблице 9.4.2.1-1.

Таблица 9.4.2.1-1 – API обнаружения интерфейса хоста ECI

Сообщение	Тип	Направление	Маркер	Описание
getApis	Get	C→H	0x0	Получить доступные интерфейсы API-хоста
getApiVersions	Get	C→H	0x1	Получить доступные версии API-хоста
setApiVersion	Set	C→H	0x2	Установить используемую версию API-хоста

9.4.2.2 Сообщение getApis

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

- Этот запрос возвращает битовый массив maxNrApis, который указывает на интерфейсы API, поддерживаемые **хостом ECI**.

Определение свойства

- Доступность интерфейса API хоста с маркером **a** и (**a <maxNrApis**) определяется как (**(*result*[a/32]>>(a%32))&0b1 == 0b1).**

Определение параметра

maxNrApis: ushort	Максимальный номер интерфейса API, для которого возвращается результат, плюс единица

9.4.2.3 Сообщение getApiVersions()

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

- Этот запрос возвращает битовый массив **maxNrVersions**, который указывает версии **api**, поддерживаемые **хостом ECI**.

Определение свойства

- Доступность версии API с маркером **api** для версии **v** и (**v <maxNrVersions**) определяется как (**(*result*[v/32]>>(v%32))&0b1 == 0b1)**.

Определение параметра

maxNrVersions: ushort	Максимальный номер версии, для которого возвращается результат, плюс единица

9.4.2.4 Сообщение setApiVersion()

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

- для параметров от **api** до **version**. Вызов следует производить только один раз (последующие вызовы не оказывают действия).

Определение параметра

api: ushort	Маркер API, для которого должна быть установлена версия
version: ushort	Номер версии **api**, который будет использоваться в следующем сеансе связи между клиентом и хостом

Подробная семантика

- Если **version** отличается от существующей версии API, поддерживаемой **api**, то должна быть задана первая из более высоких версий, поддерживаемых API, при наличии таковой; в противном случае задается максимальная версия API.
- **Клиенты ECI** проверяют доступность версии API перед выполнением ее инициирования.

ПРИМЕЧАНИЕ. – Отсутствие проверки может привести к явно непредвиденному режиму работы API или возникновению ошибок.

9.4.3 API для доступа к ресурсу пользовательского интерфейса хоста ECI

9.4.3.1 Введение

Этот раздел определяет среду прикладных программ для приложений **ECI**, позволяя **клиенту ECI** устанавливать интерфейс взаимодействия с **пользователем**. Приложения **ECI** размещаются **клиентами ECI** и выполняются на **хосте ECI**. Эти приложения используют HTML-браузер, который устанавливается в телевизионных устройствах для целого ряда платформ от поставщиков устройств и радиовещательных организаций.

На рисунке 9.4.3.1-1 показаны отдельные объекты в среде прикладных программ **ECI**. **Клиент ECI** не управляет и не связывается напрямую с запущенным им приложением **ECI**. Он использует прокси-сервер, предоставляемый **хостом ECI**. Прокси-сервер использует определяемый в пункте 9.4.3.4 интерфейс API, который позволяет **клиентам ECI** запускать и останавливать приложения **ECI**, а также связываться с запущенными приложениями **ECI**, например, для обработки данных, вводимых **пользователем**. Связь приложения **ECI** с **клиентом ECI** осуществляется прокси-сервером путем перекодирования запроса GET HTTP-браузера либо в ресурс из контейнера приложения, либо в запрос интерфейса API reqUiClientQuery **клиенту ECI**, как определяется в пункте 9.4.3.4.8. Последний может предоставить **клиенту ECI** данные, вводимые **пользователем**, и позволить **клиенту ECI** предоставить отклик, содержащий динамический контент. Контейнер приложений предоставляет (обширные) статические ресурсы для формирования экранов пользовательского интерфейса (UI). **Клиент ECI** предоставляет персонализированные входные данные для экрана интерфейса UI и принимает данные, вводимые **пользователем**.

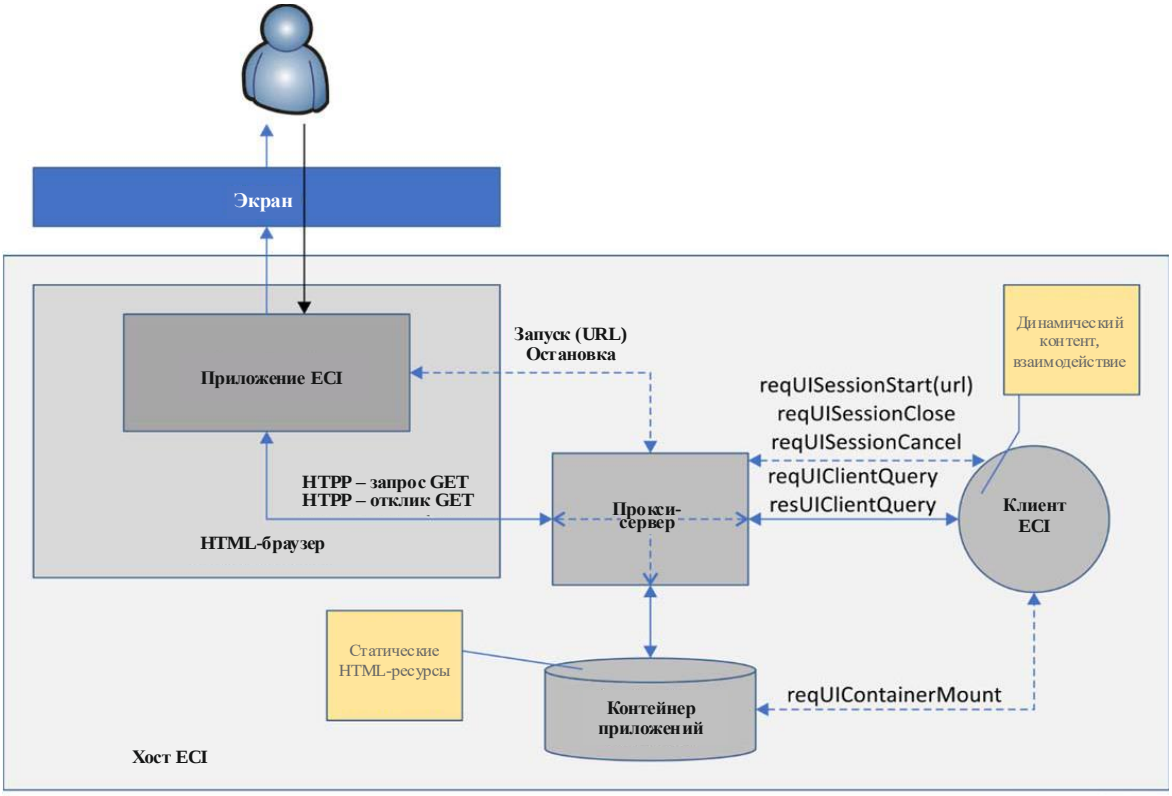


Рисунок 9.4.3.1-1 – Блок-схема пользовательского интерфейса API

9.4.3.2 Среда пользовательского интерфейса

9.4.3.2.1 Профиль браузера

Хост ECI должен обеспечивать работу HTML-браузера, который использует телевизионный профиль веб-стандартов, как определяется в [IEC 62766-5-2], в соответствии с ограничениями и расширениями, определяемыми в настоящей Рекомендации. Кроме того, этот профиль принимается системой HbbTV [b-HbbTV].

9.4.3.2.2 Ограничения

Хост ECI должен отклонять запросы HTTP к любому ресурсу сеанса **приложения ECI**, который не инициируется этим сеансом **приложения ECI**.

Адреса URL, используемые для загрузки ресурсов **приложения ECI** в браузер, представляют собой конкатенацию базового URL, уникального для сеанса, и относительного URL для обращения к **клиенту ECI** или к контейнеру приложений. Например, если базовый URL сеанса
`http://localhost:3000/session-x/`,

а ресурсом в контейнере приложений является
`main/pincode.html`

то URL-адресом браузера является
`http://localhost:3000/session-x/main/pincode.html`

При обслуживании запросов HTML-браузера **хост ECI** должен выводить тип контента ресурсов **приложения ECI** на основе расширений имен своих файлов и как минимум поддерживать:

- text/html – .html and .htm
- text/javascript – .js
- text/css – .css
- image/png – .png
- image/gif – .gif

- image/jpeg – .jpg and .jpeg

9.4.3.2.3 Функциональные возможности браузера

9.4.3.2.3.1 Модель дисплея

Окно браузера должно быть полноэкранным. Размер окна браузера должен составлять не менее 1280 × 720 пикселей. Приложение **ECI** должно быть разработано таким образом, чтобы пропорции сохранялись при масштабировании.

Графическая плоскость, на которой отображаются приложения **ECI**, должна размещаться за графической плоскостью приложений терминала и перед любой другой графической плоскостью, в том числе предназначенной для видео, субтитров и радиовещательных приложений.

Плоскость для приложений **ECI** полностью покрывает любую графическую плоскость, за исключением плоскости приложений терминала. Фон окна браузера должен быть прозрачным, то есть когда плоскость не покрывается каким-либо элементом HTML приложения **ECI**, графические плоскости, расположенные снизу (одна из которых, как правило, содержит окно видеотрансляции), должны быть видимыми. Если характеристика фонового цвета основного (body) элемента в таблице CSS имеет значение transparent (прозрачность), то фоновое окно браузера должно быть прозрачным.

Если терминалу необходимо временно перекрыть приложение **ECI**, например чтобы отобразить системное меню или информационный баннер согласно действию **пользователя**, приложение **ECI** должно терять фокус ввода. Если приложение **ECI** теряет фокус ввода, на объекте Window возникает размывка (blur).

После того как терминал закрывает свой пользовательский интерфейс, а приложение **ECI** все еще запущено, ему вновь переходит фокус ввода. Если приложение **ECI** получает фокус ввода, на объекте Window возникает событие фокус (focus). Браузер должен поддерживать формат цвета RGBA32.

9.4.3.2.3.2 Текст и шрифты

В браузер должен быть встроен пропорциональный шрифт. Приложения **ECI** могут выбирать встроенный шрифт, используя "sans-serif" или "default" в качестве типовых наименований семейства шрифтов. Набор символов, поддерживаемый встроенным шрифтом, должен соответствовать региону, в котором эксплуатируется устройство. В качестве альтернативных шрифтов и наборов символов приложения **ECI** могут использовать шрифты CSS3 Web Fonts, определяемые в [IEC 62766-5-2]. Браузер должен поддерживать как минимум один доступный для загрузки веб-шрифт для каждого приложения **ECI**.

Браузер должен поддерживать кодирование UTF-8 для всех текстовых ресурсов приложения **ECI,** то есть HTML-документов, скриптов и таблиц стилей.

9.4.3.2.3.3 Графические форматы

Браузер должен поддерживать следующие форматы графики: GIF [W3C GIF V89a], JPEG [ITU-T T.871] и PNG [W3C PNG].

9.4.3.2.3.4 Ввод данных пользователя

Браузер должен поддерживать дистанционный ввод данных **пользователя**, используя DOM3 KeyboardEvents. Когда приложение **ECI** запускается и получает фокус ввода, **хост ECI** разрешает **пользователю** инициировать следующие события:

- цифровые клавиши 0–9;
- клавиши управления курсором: вправо, влево, вверх, вниз, Enter и BrowserBack.

Поддержка устаревших атрибутов keyCode и charCode не требуется.

9.4.3.2.3.5 Хранение данных

Браузер должен обеспечивать хранение данных сеанса для API WebStorage и cookie-файлов сеанса. **Клиенту ECI** следует использовать свою внутреннюю память для хранения информации во время сессий браузера.

9.4.3.2.3.6 Доступ приложения ECI к статическим HTML-ресурсам

Прокси-сервер, получающий HTTP-запросы от **приложения ECI**, сопоставляет относительный URL (то есть расширение из базового URL-сеанса) с относительным путем в контейнер приложения, смонтированным **клиентом ECI**. Прямое сопоставление относительного URL с файлом: относительный URL directoryname1/directoryname2/.. / directorynameN/filename соотносится с именем файла в директории directorynameN, содержащейся в... содержащейся в директории directoryname2, содержащейся в директории directoryname1.

Структура директории контейнера приложения и файлы должны соответствовать следующим ограничениям:

- все имена файлов и директорий должны быть представлены буквенно-цифровых символами и символами "." (точка) и "_" (нижнее подчеркивание), количество которых не должно превышать 40.

Дополнительные требования к ресурсам или характеристикам контейнера приложения предлагаются в [b-ITU-T J Suppl. 7].

9.4.3.2.3.7 Связь клиента ECI с приложениями ECI

Браузер поддерживает запрос XmlHttpRequest, как определено в пункте 9.4.3.2.1 настоящей Рекомендации. Маршрутизация связи между приложениями **ECI** и **клиентами ECI** осуществляется через прокси-сервер **хоста ECI**. Приложение **ECI** может выполнять HTTP-запрос GET, используя запрос XMLHttpRequest API, как определяется в настоящем разделе. URL HTTP-запроса состоит из базового URL сеанса **приложения ECI**, как определяется в пункте 9.4.3.2.2, и относительного URL '/client'. Любые параметры должны быть включены в строку запроса в виде пар ключ–значение. Ключи и значения должны быть представлены только символами ASCII. Максимальная длина ключей должна составлять 31 символ, а максимальная длина значений – 255 символов.

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

После получения прокси-сервера HTTP-запроса **хост ECI** отправляет сообщение reqUiClientQuery **клиенту ECI** приложения **ECI**, содержащее выделенную строку запроса в виде пар "ключ–значение", как определяется в пункте 9.4.3.4.5. Отклик **клиента ECI**, направляемый хосту, должен включать следующий параметр:

- type – строка, соответствующая типам медиаданных, как определяется соответствующими стандартами и документально зафиксировано в базе данных типов медиаданных [b-IANA], например application/json определяется в [b-IETF RFC 8259];
- status code – целое число, применяемое в отклике на запрос GET, то есть положительный результат должен быть равен 200;
- body – строка размером не более 64 кбайтов.

На этой основе **хост ECI** формирует HTTP-отклик GET, направляемый браузеру, посредством задания параметра type для заголовка Content-Type, значения ошибки для HTTP-статуса и значения параметра body для основной части отклика.

Связь с приложениями HTML, полученными не от **клиента ECI**, выходит за рамки этой версии настоящей Рекомендации.

9.4.3.3 Жизненный цикл приложений

9.4.3.3.1 Запуск приложения ECI

Телевизионный экран относится к совместно используемым ресурсам. Его заполняют контентом приложений терминала, радиовещательных программ, приложений **операторов** и сторонних приложений. В данной версии настоящей Рекомендации определяется среда прикладных программ для

основных **пользовательских** интерфейсов, необходимых для работы модуля **ECI**, например ввод PIN, информация о подписке и т. д.

Запуск запросов **клиентов ECI**, направляемых **хостам ECI**, ограничивается следующими случаями:

- **хост ECI** намеревается начать представление мультимедийных данных (например, после настройки на радиовещательный канал), которые обрабатываются **клиентом ECI**;
- **хост ECI** представляет мультимедийные данные, которые обрабатываются **клиентом ECI**;
- **хост ECI** направляет запрос **клиенту ECI** о показе **меню приложения**;
- **клиент ECI** сообщает, что намеревается запустить **приложение ECI**, связанное с потоком, не содержащим контент; далее **хост ECI** может обеспечить, чтобы диалог происходил по запросу **пользователя** и не конфликтовал с контентом, воспроизводимым на экране: то есть отсутствует удаление, отключение или перекрытие контента, выводимого на экран третьими лицами и выбранного для просмотра **пользователем**.

Касательно вышеуказанного, запрос о запуске для выполнения взаимодействия в целях делегированной родительской аутентификации с **пользователем**, как определяется в пункте 9.8.2.11, рассматривается как запрос о запуске, инициированном **клиентом ECI**, который запросил исходную родительскую аутентификацию, как определяется в пункте 9.8.2.10.

Экранный конфликт определяется как ситуация, в которой **клиент ECI** отправляет **хосту ECI** запрос о запуске **приложения ECI** (открыть сеанс через пользовательский интерфейс), но вышеуказанные условия для запуска не выполняются.

Если **хост ECI** имеет возможность запускать интерактивные приложения, то он должен запустить по крайней мере одно **приложение ECI** во время воспроизведения такого интерактивного контента, связанного с мультимедийными данными, представленными на экране. Подобное **приложение ECI** должно быть непосредственно связано с мультимедийными данными, представленными на экране. Запуск **приложения ECI** не должен останавливать воспроизведение на экране интерактивного контента; этот контент должен быть способен возобновить взаимодействие с **пользователем** при остановке **приложения ECI**.

Хост ECI должен довести до сведения **пользователя** намерение **клиента ECI** запустить **приложение ECI**, связанное с потоком, не содержащим контент, или разрешить **клиенту ECI** запускать такое **приложение ECI** на регулярной основе без **экранного конфликта**. Данные действия могут быть выполнены, например, если такие **приложения ECI** запускаются при включении питания или в режиме ожидания, а также при определенных действиях **пользователя** в ответ на появление иконки оповещения в баннере или в меню **хоста ECI**, которое отображается постоянно. **Клиенты ECI** не должны допускать частый запуск подобных **приложений ECI** и должны ограничивать цели аспектами, влияющими на бесперебойную работу **клиента ECI**.

Запущенное **клиентом ECI приложение ECI** загружается в контекстах просмотра, которые недоступны из контекстов просмотра радиовещательных и других сторонних приложений.

Окно браузера должно быть видимым в течение одной секунды, а **приложение ECI** должно быть полностью загружено.

Будущие версии настоящей Рекомендации будут, вероятно, содержать расширенные модели жизненного цикла и механизмы разрешения конфликтов, а также допускать связь с приложениями HTML, запущенными извне.

9.4.3.3.2 Завершение приложения ECI

Для остановки **приложения ECI клиент ECI** направляет **хосту ECI** сообщение reqUISessionStop. Идентификатор uiSessionId включен в данный запрос и возвращается **хостом ECI** в отклике resUISessionOpen. Приложение **ECI** должно быть остановлено. Механизм остановки зависит от реализации системы – это может быть как закрытие, так и сворачивание окна браузера. Независимо от этого приложение **ECI** теряет фокус ввода, и браузер прекращает отправлять сообщения KeyboardEvents приложению **ECI**.

Кроме того, приложение **ECI** должно быть остановлено, если какое-либо действие **пользователя** (например, нажатие P+/P-) приводит терминал в состояние, в котором запуск приложения **ECI** запрещен. **Хост ECI** направляет **клиенту ECI** сообщение reqUiSessionCancel.

9.4.3.4 Интерфейсы API, предназначенные для связи с пользователем

9.4.3.4.1 Список сообщений API, предназначенных для связи с пользователем

Для предоставления основного пакета статических ресурсов HTML, необходимых для генерации **пользовательского** интерфейса, API **пользовательского** интерфейса разрешает **клиенту ECI** монтировать загружаемый файл контейнера приложения UI. Прокси-сервер автоматически разрешает передавать все HTTP-запросы, не направленные клиенту, из браузера в файл контейнера приложения.

Хост ECI может предложить **клиенту ECI** запустить приложение либо в отклике на запрос **пользователя** о предоставлении **клиенту ECI** доступа к **меню приложения**, либо путем уведомления **клиента ECI** об отсутствии конфликтов, препятствующих представлению **пользователю приложения ECI**, не связанного с **указателем медиаданных**, с помощью сообщения reqUiSessionCommence. **Клиент ECI** может уведомить о заинтересованности в запуске подобного диалога, не связанного с **указателем медиаданных**, посредством сообщения setUiClientAttention. По существу, при отсутствии **экранного конфликта** данная процедура позволяет устанавливать связь с более низким приоритетом между **клиентом ECI** и **пользователем**.

Клиент ECI открывает все сеансы **пользовательского** интерфейса посредством сообщения reqUiSessionOpen. Относительный URL, воспроизводящий первый экран пользовательского интерфейса, предоставляется в качестве параметра. **Клиент ECI** и **хост ECI** могут завершить сеанс **пользовательского** интерфейса посредством сообщений reqUiSessionClose и reqUiSessionCancel соответственно.

Сообщение reqUiClientQuery позволяет **приложению ECI** в браузере отправлять запросы, содержащие параметры, через прокси-сервер **клиенту ECI**, который затем может отправлять в ответ данные для HTML-приложения. Это соединение позволяет **приложению ECI** представлять конкретные данные для **клиента ECI** и обеспечивать ввод данных **пользователя** для **клиента ECI** тем же способом, каким HTML-приложение взаимодействует с динамическим HTTP-сервером.

Список всех интерфейсов API, определяемых в данном разделе, приведен в таблице 9.4.3.4.1-1.

Таблица 9.4.3.4.1-1 – Сообщения API пользовательского интерфейса

Сообщение	Тип	Направление	Маркер	Описание
reqUiContainerMount	A	C→H	0x0	Монтирует контейнер приложения UI с ресурсами HTML для поддержки сеансов через UI
setUiClientAttention	S	C→H	0x1	**Клиент ECI** сообщает о намерении начать сеанс через UI, не связанный с **указателем медиаданных**
reqUiSessionCommence	A	H→C	0x2	**Хост ECI** предлагает **клиенту ECI** открыть сеанс через UI
reqUiSessionOpen	A	C→H	0x3	Клиент **ECI** отправляет запрос на открытие сеанса связи с **пользователем** через **пользовательский** интерфейс и отображение контента на экране
reqUiSessionClose	A	C→H	0x4	**Клиент ECI** завершает сеанс через **пользовательский** интерфейс
reqUiSessionCancel	A	H→C	0x5	**Хост ECI** отменяет сеанс через **пользовательский** интерфейс
reqUiClientQuery	A	H→C	0x6	**Клиент ECI** принимает запрос от HTML-приложения в браузере и отправляет (динамический) отклик

9.4.3.4.2 Сообщение reqUiContainerMount

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

- Это сообщение разрешает **клиенту ECI** давать указание **хосту ECI** о назначении файла в качестве контейнера приложения **клиента ECI**, содержащего ресурсы HTML для своего **приложения ECI**. В случае положительного результата возвращается содержимое файла EciIndex.txt в главной директории контейнера приложения.

Определения параметра запроса

filename: fileName	Имя файла в файловой системе клиента ECI, которая будет назначена в качестве контейнера приложения
pk: PubKey	Открытый ключ для проверки подписи контейнера приложения

Определения параметра отклика

indexFileLen: uint	Длина файла индекса
indexFile: uchar	Содержание файла индекса

Подробная семантика

- Используемые далее квадратные скобки ([и]) с подробным текстом между ними обозначают разграничение полей и структур в контейнера ZIP-файлов.
- Подпись для проверки файла контейнера находится в поле [.ZIP file comment] структуры [end of central directory record] (см. спецификацию формата zip-файла версии 6.3.3 PKWARE® Inc., как указывается в [ISO/IEC 21320].
- [.ZIP Comment Field] определяется как *окончание* следующей строки, состоящей из всех символов ASCII: 'ECI_SIGNATURE ="', за которым следует значение структуры ECI_Data_Signature, как определяется в таблице 6.3-3, закодированной в виде шестнадцатеричной строки с использованием прописных букв, за которыми следуют "" (закрывающие скобки).

ПРИМЕР:

ECI_SIGNATURE="01000000FCB1F60456719035"

Длина закодированной строки подписи данных для ECI_Data_Signature 1-го типа составляет 520 символов.

- **Хост ECI** проверяет подпись, используя параметр открытого ключа pk и процесс вычисления подписей, определенный в пункте 5.6. Подпись вычисляется по файлу контейнера до поля [Zip comment field] в структуре [end of central directory record]; при этом значение поля [.Zip comment length field] задается равным 0x0000.
- Файл индекса определяется как файл с названием "EciIndex.txt" в главной директории файла контейнера.

- **Клиент ECI** монтирует действующий контейнер приложения пользовательского интерфейса, если это необходимо для сеансов через пользовательский интерфейс.
- **Клиент ECI** должен быть способен отображать для **пользователя** базовое аварийное сообщение в случае ошибки загрузки и монтировать контейнер приложения пользовательского интерфейса.

Примечания по приложению

- Клиенты могут загружать файлы контейнера приложения в свою файловую систему либо с онлайн-сервера, используя API HTTP (S) (см. пункт 9.4.4.6), либо из радиовещательного транспортного потока, используя API карусели передачи данных.
- Файл "EciIndex.txt" может содержать информацию о версии для пользовательского интерфейса, проверенную с помощью подписи с открытым ключом.

Коды ошибок, касающиеся сообщения reqUiContainerMount, определены в таблице 9.4.3.4.2-1.

Таблица 9.4.3.4.2-1 – Коды ошибок reqUiContainerMount

Название	Описание
ErrUiContainerFileNot	См. таблицу 9.4.3.4.9-1
ErrUiContainerNot	
ErrUiContainerSignature	
ErrUiContainerIndexTxtNot	

9.4.3.4.3 Сообщение setUiClientAttention

C→H setUiClientAttention(uint **clientAttention**)

- Это сообщение указывает на намерение клиента ECI открыть сеанс связи с пользователем через пользовательский интерфейс. Этот сеанс не имеет отношения к указателю медиаданных (тип UI-сеанса соответствует EciUiSessionDiaReq, см. пункт 9.4.3.4.4).

Определение показателей

clientAttention: uint	Определяются следующие значения: 0x0 – внимание пользователя не требуется; 0x1 – внимание пользователя желательно. Все другие значения резервируются

Постусловия

- Если clientAttention = 0x0, то сообщения reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq) будут выдаваться **хостом ECI**.
- Если clientAttention = 0x1, то сообщение reqUiClientSessionCommence(uiSessionType=EciUiSessionDiaReq) будет выдаваться **хостом ECI** при условии отсутствия ожидающих сообщений данного типа.

9.4.3.4.4 Сообщение reqUiSessionCommence

H→C reqUiSessionCommence (uint **uiSessionType**) →
C→H resUiSessionCommence ()

- Это сообщение позволяет **хосту ECI** предлагать **клиенту ECI** открыть сеанс связи определенного типа через пользовательский интерфейс.

Определения параметра запроса

uiSessionType: uint	Имя файла в файловой системе **клиента ECI**, которая будет назначенным контейнером приложения. Конкретные значения определяются в таблице 9.4.3.4.4-1. Допускаются только значения EciUiSessionAppMenu и EciUiSessionDiaReq

Таблица 9.4.3.4.4-1 – Типы сеансов пользовательского интерфейса ECI

Название	Значение	Описание
EciUiSessionDiaReq	0x00	**Клиент ECI** запросил сеанс связи с конечным пользователем через пользовательский интерфейс посредством сообщения setUiClientAttention (без привязки к определенному **указателю медиаданных**), а **хост ECI** может предоставить в этой связи сообщения reqUISessionOpen от **клиента ECI**
EciUiSessionAppMenu	0x01	**Меню приложения клиента ECI**. Тем самым обеспечивается инициированный **пользователем** доступ ко всем необходимым настройкам, информации и функциям, которые могут быть инициированы **пользователем**
EciUiSessionMh	0x02	**Клиент ECI** запросил сеанс через пользовательский интерфейс в связи с процедурами для **указателя медиаданных**
EciUiSessionParAuthDel	0x03	**Клиент ECI** запросил сеанс через пользовательский интерфейс в целях выполнения диалога делегированной аутентификации родительской зоны для обработки контента по **указателю медиаданных**
RFU	Прочее	Зарезервировано для использования в будущем

ПРИМЕЧАНИЕ. – Значения в таблице 9.4.3.4.4-1 определяются в предложенном порядке приоритета. На основе такого порядка могут быть даны предложения по урегулированию конфликтов фокуса пользовательского интерфейса в структуре **хоста ECI**.

Подробная семантика

- **Клиент ECI** должен быть способен представлять **меню приложения**. **Меню приложения** должно, как минимум, позволять **пользователю** проверять версию **клиента ECI**, ссылку на **систему управления платформой** и режим работы **клиента ECI**.

Предварительные условия (запрос)

- Для **сеанса** через пользовательский интерфейс не должно предварительно выдаваться ожидающее сообщение reqUiSessionCommence **клиенту ECI**.

Постусловия (отклик)

- **Клиент ECI** должен выдавать сообщение reqUiSessionOpen с соответствующим типом сеанса через пользовательский интерфейс или сообщать об ошибке.

Коды ошибок, относящиеся к сообщению reqUiSessionCommence, определяются в таблице 9.4.3.4.4-2.

Таблица 9.4.3.4.4-2 – Коды ошибок reqUiClientSessionCommence

Название	Описание
ErrUiResourceError	См. таблицу 9.4.3.4.9-1
ErrUiClientError	

9.4.3.4.5 Сообщение reqUiSessionOpen

C→H reqUiSessionOpen(uint **uiSessionType,** ushort **mH,** uint **relUrlLen,** char **relUrl[]) →**
H→C resUiSessionOpen(ushort **uiSessionId)**

- Это сообщение позволяет **клиенту ECI** запрашивать новый сеанс через пользовательский интерфейс у **хоста EC**.

Определения параметров запросов

uiSessionType: uint	Тип сеанса через пользовательский интерфейс, определяемый в таблице 9.4.3.4.4-1. Параметр mH будет значимым, только если значение равно EciUiSessionMh или EciUiSessionParAuthDel
mH: ushort	Указатель медиаданных сеанса обработки контента, с которым связан интерфейс MMI
relUrlLen: uint	Длина relUrl в байтах
relUrl: char[]	Относительный URL, завершается нулевым символом. Добавленный к сеансу базовый URL будет формировать URL-адрес браузера для запуска сеанса через пользовательский интерфейс. См. пункт 9.4.3.2.2.

Определения параметра отклика

uiSessionId: ushort	ID нового сеанса через пользовательский интерфейс

Подробная семантика

- **Клиент ECI** должен быть способен обрабатывать несколько сеансов через пользовательский интерфейс одновременно. Однако обязательной является поддержка лишь одного сеанса через пользовательский интерфейс типа EciUiSessionAppMenu или EciUiSessionAppMenu; кроме того, требуется не более одного сеанса через пользовательский интерфейс типа EciUiSessionMh для каждого открытого **указателя медиаданных**.
- **Клиент ECI** должен иметь возможность открывать одновременные сеансы через пользовательский интерфейс типа EciUiSessionMh.
- **Клиент ECI** должен быть способен открывать одновременные сеансы через пользовательский интерфейс типа EciUiSessionParAuthDel, если **клиент ECI** поддерживает API делегирования аутентификации родительской зоне. Такие сеансы через пользовательский интерфейс могут осуществляться параллельно с другими сеансами через пользовательский интерфейс **клиента ECI**.
- **Хост ECI** может поддерживать один или несколько одновременных сеансов через пользовательский интерфейс в соответствии с режимами приложения устройств **CPE**.

Предварительные условия (запрос)

1) Если значение uiSessionType равно EciUiSessionAppMenu или EciUiSessionDiaReq, то этому сообщению должно предшествовать сообщение reqUiClientCommence с тем же параметром uiSessionType.
2) Если значение uiSessionType равно EciUiSessionParAuthDel, то этому сообщению должно предшествовать сообщение reqParAuthDel для параметра mH указателя медиаданных от **хоста ECI клиенту ECI**.
3) Если значение uiSessionType равно EciUiSessionMh, параметр Mh должен обозначать открытый сеанс указателя медиаданных.

Предварительные условия (отклик)

1) Если значение uiSessionType равно EciUiSessionAppMenu, или EciUiSessionDiaReq, или EciUiSessionParAuthDel, **хост ECI** принимает только тот запрос сеанса через пользовательский интерфейс, который он направлял ранее, и только в том случае, если его причина не устранена, а его статус не приведет к **экранному конфликту**.
2) Если значение uiSessionType равно EciUiSessionMh, то **хост ECI** предоставляет запрос сеанса через пользовательский интерфейс, если он может установить целенаправленное взаимодействие с **пользователем**, не приводящее к экранному конфликту приоритетов.
3) **Хосты ECI** не должны отклонять второй сеанс от **клиента ECI**, если значение типа второго сеанса uiSessionType равно EciUiSessionParAuthDel. **Хосту ECI** разрешается отменять первый сеанс.

Замечания по применению

1) Если для записи используется сеанс **указателя медиаданных** и нет возможности инициировать диалог с пользователем по причине возможного **экранного конфликта** или отсутствия активного экрана, **хост ECI** должен отказаться от сеанса.

2) Приложениям **хоста ECI** рекомендуется размещать сеансы через пользовательский интерфейс для аутентификации родительской зоны, например, при программировании будущих записей, которые могут потребовать аутентификацию родительской зоны, используя сообщение reqParAuthCid интерфейса API аутентификации родительской зоны (см. пункт 9.8.2.10).

3) Хосты ECI могут отменять сеанс связи с клиентом ECI через пользовательский интерфейс для того, чтобы разрешить новый сеанс со значением uiSessionType, равным EciUiSessionParAuthDel или EciUiSessionMh.

Коды ошибок, относящиеся к сообщению reqUiSessionOpen, определяются в таблице 9.4.3.4.5-1.

Таблица 9.4.3.4.5-1 – Коды ошибок reqUiClientSessionStart

Название	Описание
ErrUiScreenConflict	См. таблицу 9.4.3.4.9-1
ErrUiNoScreen	

9.4.3.4.6 Сообщение reqUiSessionClose

C→H reqUiSessionClose(ushort **uiSessionId) →**
H→C resUiSessionClose(ushort **uiSessionId)**

- Это сообщение позволяет **клиенту ECI** закрыть существующий сеанс через пользовательский интерфейс.

Определение параметра запроса

uiSessionId: ushort	ID сеанса через пользовательский интерфейс, подлежащего закрытию

Определение параметра отклика

uiSessionId: ushort	ID сеанса через пользовательский интерфейс, который был закрыт

Предварительные условия (запрос)

1) Сеанс через пользовательский интерфейс с ID uiSessionId должен быть открыт.

2) Дополнительные сообщения, относящиеся к uiSessionId, хосту ECI не отправляются.

Предварительные условия (отклик)

1) Дополнительные сообщения, относящиеся к uiSessionId, **клиенту ECI** не отправляются.

9.4.3.4.7 Сообщение reqUiSessionCancel

H→C reqUiSessionCancel (ushort **uiSessionId,** uint **reason) →**
C→H resUiSessionCancel (ushort **uiSessionId)**

- Это сообщение позволяет **хосту ECI** закрыть существующий сеанс связи с **клиентом ECI** через пользовательский интерфейс. Это сообщение предназначено для использования **хостом ECI** в случаях, когда условия для отображения **приложения ECI** больше не соблюдаются, например если **пользователь** переключается на другой канал, принадлежащий другому **клиенту ECI**, что приводит к **экранному конфликту**.

Определения параметра запроса

uiSessionId: ushort	ID сеанса через пользовательский интерфейс, который должен быть отменен
reason: uint	Основание для отмены сеанса. Значения определяются в таблице 9.4.3.4.9-1.

Определения параметра отклика

uiSessionId: ushort	ID сеанса через пользовательский интерфейс, который был отменен

Предварительные условия (запрос)

1) Сеанс с ID uiSessionId должен быть открыт.

2) Дополнительные сообщения со ссылкой на uiSessionId не отправляются.

Предварительные условия (отклик)

1) Дополнительные сообщения со ссылкой на uiSessionId не отправляются.

9.4.3.4.8 Сообщение 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[])**

- Это сообщение передает HTTP-запрос посредством **приложения ECI**, запускаемого в браузере **хоста ECI**, как описывается в пункте 9.4.3.2.3.7, и разрешает **клиенту ECI** отправлять обратный HTTP-отклик **приложению ECI**.

Определения параметра запроса

uiSessionId: ushort	Id сеанса через пользовательский интерфейс, из которого посылается запрос
queryLen: uint	Длина параметра запроса в байтах
query[]: KeyValPair	Содержит пары ключ–значение параметров HTTP-запроса, отправляемого браузером

Определения типа для KeyValPair

```
#define MaxKeyLen 32
#define MaxValLen 256

typedef struct KeyValPair {
    char key[MaxKeyLen];    /* Ключ пары ключ-значение, завершающийся нулевым символом*/
    char val[MaxValLen];    /* Значение пары ключ-значение, завершающееся нулевым символом */
} KeyValPair;
```

Определения параметра отклика

uiSessionId: ushort	Id сеанса через пользовательский интерфейс
statusCode: uint	HTTP-код статуса, как определено в [IETF RFC 7231]
typeLen: uint	Длина параметра type в байтах
type[]: char	Тип отклика в виде строки символов ASCII, завершающейся нулевым символом
bodyLen: uint	Длина параметра body в байтах
body[]: uchar	Сообщение-отклик HTTP

Предварительные условия (запрос)

1) Идентификатор **uiSessionId** открыт.

Подробная семантика

- В случае неверно отформатированной строки запроса из **приложения ECI хост ECI** может вернуть код 400 статуса HTTP и не посылать запрос **клиенту ECI**.
- Взаимосвязь параметра сообщения с HTTP-запросом и откликом от браузера определяется в пункте 9.4.3.2.3.7.

9.4.3.4.9 Коды ошибок для API-связи с пользователем

Коды ошибок, относящиеся к связи через интерфейс **пользователя**, перечисляются в таблице 9.4.3.4.9-1.

Таблица 9.4.3.4.9-1 – Коды ошибок API-связи с пользователем

Название	Значение	Описание
ErrUiContainerFileNot	–256	Файл контейнера приложения пользовательского интерфейса не найден
ErrUiContainerNot	–257	Файл не является действительным файлом контейнера приложения пользовательского интерфейса
ErrUiContainerSignature	–258	Сбой проверки подписи в файле контейнера приложения
ErrUiContainerIndexTxtNot	–259	Файл "EciIndex.txt" в корневой директории контейнера приложения отсутствует
ErrUiResourceError	–260	**Клиент ECI** не может смонтировать ресурс контейнера приложения пользовательского интерфейса
ErrUiClientError	–261	Режим работы **клиента ECI** не позволяет представить пользовательский интерфейс
ErrUiDiaNoMore	–262	Запрос диалога от **клиента ECI** более не действителен
ErrUiScreenConflict	–263	**Хост ECI** имеет **экранный конфликт** и не может размещать или поддерживать сеанс
ErrUiNoScreen	–264	Доступ **хоста ECI** к экрану для представления сеанса через пользовательский интерфейс отсутствует или потерян
RFU	Прочее	Файл контейнера приложения пользовательского интерфейса не найден

9.4.4 Интерфейс API для доступа к ресурсу IP-стека хоста ECI

9.4.4.1 Введение

В устройствах **CPE**, оснащенных IP-стеком, **хост ECI** предоставляет услугу доступа в интернет для **клиентов ECI**. **Клиенты ECI** могут отправлять сообщения по протоколу UDP/IP и открытым TCP/IP-соединениям одноранговым узлам как в режиме **клиента ECI**, так и в режиме сервера, используя **хосты ECI**. Имена **хостов ECI** могут быть преобразованы в IP-адреса с помощью доступных услуг DNS в **хосте ECI**.

Предоставляемые услуги защищены лишь в рамках общей безопасности программного обеспечения **CPE**. Таким образом, если программное обеспечение **CPE** за пределами **хоста ECI** окажется под угрозой, любой IP-трафик может быть подвергнут вмешательству.

Интерфейс API **клиента ECI** для IP-соединений основан на парадигме BSD-сокета, которая используется во многих современных операционных системах.

Определение интерфейса API разделено на четыре части.

1) Основные IP-сокеты ECI и набор функций DNS (пункт 9.4.4.3).
2) Связь по UDP/IP с использованием IP-сокета ECI (пункт 9.4.4.4).
3) Связь по TCP/IP с использованием IP-сокета ECI (пункт 9.4.4.5).
4) Связь по протоколу HTTP(S) с использованием HTTP-услуг **хоста ECI** (пункт 9.4.4.6).

9.4.4.2 Базовые спецификации

Хост ECI, поддерживающий IP-соединение, должен использовать IP-протокол [IETF RFC 791], в том числе протокол IPv6 [IETF RFC 8200], с соответствующими обновлениями. Он должен предоставлять средство преобразования имен **хоста ECI** в IP-адреса с использованием службы DNS согласно [IETF RFC 1034], [IETF RFC 1035] и соответствующим обновлениям.

Хост ECI предоставляет простой краткий протокол передачи ненадежных сообщений, поддерживая протокол UDP через IP в соответствии с [IETF RFC 768], включая соответствующие обновления. **Хост ECI** обеспечивает обмен надежными сообщениями через соединение с управлением потоками, поддерживая протокол TCP через IP согласно [IETF RFC 793] и соответствующим обновлениям.

Хост ECI не обязан обеспечивать поддержку многоадресной рассылки по протоколу UDP ни в режиме передачи, ни в режиме приема.

9.4.4.3 IP-сокеты интерфейса ECI

9.4.4.3.1 Общие сведения

Клиенты ECI могут открывать IP-сокеты **ECI** для связи в режимах передачи и приема с использованием протоколов TCP и IP.

ПРИМЕЧАНИЕ. – Термин "сокет" предполагает сходство с оригинальными BSD-сокетами, которые используются во многих операционных системах. IP-сокеты **ECI** аналогичны по концепции, но обладают рядом особых свойств, отличающих их от BSD-сокетов. К ним относится, в частности, полностью асинхронный режим работы.

IP-сокеты **ECI** являются конечными точками для IP-связи. **Клиенты ECI** могут открывать сокет, определяя локальный номер порта и готовность принимать **запросы** на входящие соединения (функционируя как сервер TCP/IP). Такие сокеты могут закрываться, и в этом случае любые связанные с ним соединения или действия сервера также закрываются. Имя однорангового хоста может быть преобразовано в IP-адрес при помощи услуг DNS **хоста ECI**.

Перечень применимых сообщений приводится в таблице 9.4.4.3.1-1.

Таблица 9.4.4.3.1-1 – Сообщения IP-сокета

Тип	Сообщения	Напр.	Маркер	Описание
reqIpSocket	A	C→H	0x0	Открывает IP-сокет **ECI**
reqIpClose	A	C→H	0x1	Закрывает IP-сокет **ECI**
reqIpAddrinfo	A	C→H	0x2	Получает адрес (удаленного) **хоста ECI**

Определения типа структуры для этих интерфейсов API приводятся в пункте 9.3.

Определения типа для API IP-сокета

```
typedef struct Addrinfo {
   ushort addressType;          /* адрес IPv4 или IPv6 */
   uchar  ipAddress[16];        /* IP-адрес */
   ushort port;                 /* номер порта, если это применимо */
} Addrinfo;
```

Определения полей

addressTyp: ushort.	См. таблицу 9.4.4.3.4-1. Допускаются только значения ProtPrefIPv4 или ProtPrefIPv6. Это поле определяет длину hostAddress в виде 4 или 16 байтов (см. примечание)
ipAddress: uchar[16]	Поле 4 или 16 байтов, побайтово представляющее (в сетевом порядке) IPv4 или IPv6-адреса соответственно. Для адресов IPv4 используются первые 4 байта этого параметра
port: ushort	Номер порта сокета для соединения (поле может не использоваться)
ПРИМЕЧАНИЕ. – **ProtPrefIPv4** или **ProtPrefIPv6** определяются в таблице 9.4.4.3.4-1.	

9.4.4.3.2 Сообщение reqIpSocket

C→H reqIpSocket(uchar **source,** ushort **sourcePort,** ushort **protocol)** →
H→C resIpSocket(uchar **socketId)**

- Данное **сообщение** открывает сокет для связи по TCP или UDP на локальном IP-адресе и порте.

Определения параметра запроса

source: uchar	См. таблицу 9.4.4.3.2-1. Указывается IP-адрес хоста ECI, используемый для локального сокета (предпочтительный вариант при использовании нескольких IP-адресов). При невозможности идентификации конкретного IP-адреса хост ECI выбирает подходящий альтернативный вариант
sourcePort: ushort.	Адрес порта конечной точки локального IP-соединения. Значение, равное 0x0000, означает, что хост ECI выделяет для сокета свободный адрес порта. Другие значения, меньшие 1024, не допускаются
Protocol: ushort	См. таблицу 9.4.4.3.2-2. Указывается протокол, используемый для сокета. Особым образом выбирается протокол IPv4 или IPv6

Таблица 9.4.4.3.2-1 – Параметр источника IP

Название	Значение	Описание
IpSourceAny	0x00	IP-адрес **хоста ECI** по умолчанию
IpSourceWan	0x01	IP-адрес **хоста ECI,** используемый для (интернет) связи по сети WAN
IpSourcePriv	0x02	IP-адрес **хоста ECI,** используемый для частного IP-трафика по каналу с проприетарным IP-протоколом
IpSourceLan	0x03	IP-адрес **хоста ECI**, используемый для соединения по локальной сети
RFU	Прочее	Зарезервировано для использования в будущем

Таблица 9.4.4.3.2-2 – Параметр IP-протокола

Название	Значение	Описание
SockProtUdpIPv4	0x0001	UDP/IP с использованием Ipv4
SockProtUdpIPv6	0x0002	UDP/IP с использованием Ipv6
SockProtUdpIPany	0x0003	UDP/IP с использованием IPv4 или v6
SockProtTcpClientIpv4	0x0005	TCP/IP с использованием Ipv4, режим клиента (только для инициирования соединений)
SockProtTcpClientIpv6	0x0006	TCP/IP с использованием Ipv6, режим клиента (только для инициирования соединений)
SockProtTcpClientIpany	0x0007	TCP/IP с использованием Ipv4 или Ipv6, режим клиента (только для инициирования соединений)
SockProtTcpServerIpv4	0x0009	TCP/IP с использованием Ipv4, режим сервера (для приема входящих соединений)
SockProtTcpServerIpv6	0x000A	TCP/IP с использованием Ipv6, режим сервера (для приема входящих соединений)
SockProtTcpServerIpany	0x000B	TCP/IP с использованием Ipv4 или Ipv6, режим сервера (для приема входящих соединений)
RFU	Прочее	Зарезервировано для использования в будущем

Определения параметра отклика

SocketId: uchar.	Идентификатор открытого сокета

Семантическое описание

- Сразу после инициирования **отклик** может быть приостановлен до того момента, когда будет успешно завершено инициирование IP-адреса **хоста ECI**. Эксплуатационные характеристики предлагаются в [b-ITU-T J Suppl. 7].

Предварительные условия (запрос)

1) Максимальное количество сокетов, которые может запрашивать **клиент ECI,** не должно превышаться.

2) Действительная конфигурация параметров включает в себя источник, sourcePort и протокол.

Постусловия (отклик)

1) Сокет открывается, или в **отклике** возвращается ошибка.

Коды ошибок, связанных с открытием сокетов, перечисляются в таблице 9.4.4.3.2-3.

Таблица 9.4.4.3.2-3 – Коды ошибок resIpSocket

Название	Описание
ErrIpSourceProt	См. таблицу 9.4.4.7-1
ErrIpNoSockets	
ErrIpProtNotAvail	
ErrIpPortNotAvail	

9.4.4.3.3 Сообщение reqIpClose

C→H reqIpClose(uchar **socketId)** →
H→C resIpClose(uchar **socketId)**

- Закрывает IP-сокет и все связанные с ним соединения; все сообщения, ожидающие отправки и приема сокетом, могут быть потеряны.

Определения параметра запроса

socketId: uchar	Идентификатор сокета, подлежащего закрытию

Определения параметра отклика

socketId: uchar	Идентификатор сокета, который был закрыт

Семантическое описание

- Это **запрос** закрывает сокет и все связанные с ним IP-соединения, оставляя на усмотрения **хоста ECI** отправку одноранговым узлам необходимых сообщений о разрыве соединения (если применимо); при этом успешное завершение данного действия не является необходимым для отправки **запроса**. Сокет без связанного подключения также будет закрыт.

Предварительные условия

1) Сокет существует и находится в открытом состоянии.

Постусловия

2) Сокет закрывается и больше не может использоваться для любых соединений (пока не будет повторного назначения на reqIpSocket).

Коды ошибок, связанных с закрытием сокетов, перечислены в таблице 9.4.4.3.3-1.

Таблица 9.4.4.3.3-1 – Коды ошибок resIpClose

Название	Описание
ErrIpSocketNotOpen	См. таблицу 9.4.4.7-1

9.4.4.3.4 Сообщение reqIpAddrInfo

C→H reqIpAddrinfo(uint **hostnameLenth,** char **hostname[],** uchar **protPref)** →
H→C resIpAddrinfo(Addrinfo **ipaddress)**

- Это сообщение предоставляет информацию по IP-адресу **хоста ECI** с использованием приоритетного протокола (protPref), возвращающего адрес **хоста ECI**. В случае необходимости данный протокол использует услуги DNS **хоста ECI** для разрешения **запроса**.

Определения параметра запроса

hostNameLength: uint	Длина поля имени (в байтах)
hostname: char[]	Имя хоста IP, подлежащее разрешению; запись адреса IPv4 с точечной нотацией [IETF RFC 952], запись адреса IPv6 с нотацией двоеточиями [IETF RFC 8200] или фактическое имя хоста [IETF RFC 1123]
protPref: uchar	Указывает на приоритет IP-протокола, как определяется в таблице 9.4.4.3.4-1

Таблица 9.4.4.3.4-1 – Параметр приоритета IP-протокола

Название	Значение	Описание
ProtPrefIpv4	0x1	Должен возвращаться адрес IPv4
ProtPrefIPv6	0x2	Должен возвращаться адрес IPv6
ProtPrefAny	0x3	Должен возвращаться либо адрес IPv4, либо адрес IPv6
RFU	Прочее	Зарезервировано для использования в будущем

Определения параметра отклика

Ipaddress: Addrinfo	IP-адрес **хоста ECI**. Поле порта не определяется

Семантическое описание

- Этот запрос преобразует предоставленное имя хоста в двоичный адрес, используя услуги DNS хоста ECI. Временное отсутствие доступа к услуге DNS может вызывать задержки (например, при введении в эксплуатацию оборудования CPE); хост ECI должен обеспечить соблюдение необходимого времени ожидания (то есть **отклик** всегда принимается **клиентом ECI**).

Постусловия (отклик)

1) Разрешенный адрес хоста или ошибка.

Коды ошибок, связанных с закрытием сокетов, перечисляются в таблице 9.4.4.3.4-2.

Таблица 9.4.4.3.4-2 – Коды ошибок resIpAddrInfo

Название	Описание
ErrIpHostUnknown	См. таблицу 9.4.4.7-1
ErrIpHost	
ErrDnsOffline	

9.4.4.4 UDP/IP интерфейса ECI

9.4.4.4.1 Общие сведения

Клиенты ECI отправляют и принимают датаграммы UDP, используя открытый сокет UDP/IP **ECI**. Соответствующие сообщения определяются в таблице 9.4.4.4.1-1.

Таблица 9.4.4.4.1-1 – Сообщения сокета UDP/IP

Сообщение	Тип	Направление	Маркер	Описание
reqIpUdpSendMsg	A	C→H	0x3	Отправляет сообщение на одноранговый порт UDP
reqIpUdpRecvMsg	A	C→H	0x4	Принимает сообщение от одноранговый порта UDP

9.4.4.4.2 Сообщение reqIpUdpSendMsg

C→H reqIpUdpSendMsg(uchar **socketId,** Addrinfo **peer,** uint **datagramLength,** byte **datagram[]) → H→C resIpUpdSendMsg(**uchar **socketId)**

- Это сообщение отправляет датаграмму UDP на одноранговый узел (IP-адрес, IP-порт).

Определения параметра запроса

socketId: uchar	Длина поля имени (в байтах)
peer: Addrinfo	Одноранговый узел (IP-адрес, номер IP-порта) – получатель датаграммы
datagramLength: uint	Длина датаграммы (в байтах)
datagram: byte[]	Содержание датаграммы (байты в сетевом порядке)

Определения параметра отклика

socketId: uchar	Сокет, на котором был создан соответствующий **запрос**

Семантическое описание

- Датаграмма отправлена на одноранговый узел с использованием протокола UDP, IP-адреса хоста и порта сокета.

Предварительные условия (запрос)

1) Сокет открыт для UDP с использованием структуры адреса, аналогичной структуре однорангового узла.

Постусловия

1) Датаграмма отправляется (но может быть потеряна).

Коды ошибок, связанных с отправкой датаграмм UDP, перечисляются в таблице 9.4.4.4.2-1.

Таблица 9.4.4.4.2-1 – Коды ошибок resIpUdpSendMsg

Название	Описание
ErrIpUdpProtMismatch	См. таблицу 9.4.4.7-1
ErrIpUdpSocketNot	
ErrIpUdpTooLong	
ErrIpUdpIpOffline	

9.4.4.4.3 Сообщение reqIpUdpRecvMsg

C→H reqIpUdpRecvMsg(uchar **socketId)** →
H→C resIpUdpRecvMsg(uchar **socketId,** Addrinfo **peer,** uint **datagramLength,** byte **datagram[])**

- Это сообщение позволяет **клиенту ECI** отправлять **хосту ECI** запрос на получение датаграммы UDP от однорангового узла (то есть имя хоста и порт), отправленной на сокет с идентификатором **SocketId**.

Определения параметра запроса

socketId: uchar	Сокет (обозначающий номер порта и адрес хоста), на котором предполагается получение UDP-датаграммы

Определения параметра отклика

socketId: uchar	Длина поля имени (в байтах)
peer: Addrinfo	IP-адрес + номер порта источника датаграммы (однорангового узла)
datagramLength: uint	Длина датаграммы (в байтах)
datagram: byte[]	Содержание датаграммы (байты в сетевом порядке)

Семантическое описание

- Датаграмма может быть получена на сокете, в этом случае возвращается **отклик**.

ПРИМЕЧАНИЕ 1. – При закрытии сокета отменяются все ожидающие запросы **reqIpUdpRecvMsg**.

ПРИМЕЧАНИЕ 2. – Отправка нескольких сообщений **reqIpUdpRecvMsg** перед получением соответствующих **откликов** на одном и том же сокете разрешена, однако **хост ECI** не обязан поддерживать постановку в очередь более чем пяти подобных **запросов**.

Предварительные условия (запрос)

- Сокет открыт для UDP.

Постусловия (отклик)

- Датаграмма отправляется (но может быть потеряна).

Коды ошибок, связанные с приемом датаграмм UDP, перечисляются в таблице 9.4.4.4.3-1.

Таблица 9.4.4.4.3-1 – Коды ошибок resIpUdpRecvMsg

Название	Описание
ErrIpUdpSocketNot	См. таблицу 9.4.4.7-1

9.4.4.5 TCP/IP интерфейса ECI

9.4.4.5.1 Общие сведения

Клиенты ECI могут отправлять и получать сообщения через соединение по TCP/IP, открываемое при создании сокета, формируя эффективную двунаправленную последовательность потоков байтов, не содержащую ошибок, от локального **клиента ECI** до удаленной одноранговой услуги и обратно. Тем самым **клиент ECI** может действовать как сервер, направляющий **запросы** каналов связи от других участников (как правило, для приложений локальной сети). Соответствующие сообщения перечисляются в таблице 9.4.4.5.1-1.

Таблица 9.4.4.5.1-1 – Сообщения сокета TCP/IP

Сообщение	Тип	Направление	Маркер	Описание
reqIpTcpConnect	A	C→H	0x5	Клиент TCP подключается к одноранговому узлу сервера TCP
reqIpTcpSend	A	C→H	0x6	Направляет данные подключенному одноранговому узлу
reqIpTcpRecv	A	C→H	0x7	Получает данные от подключенного одноранговой узла
reqIpTcpAccept	A	C→H	0x8	Одноранговый узел сервера TCP принимает подключение от одноранговой узла клиента TPC

9.4.4.5.2 Сообщение reqIpTcpConnect

C→H reqIpTcpConnect(uchar **socketId,** Addrinfo **peer)** →
H→C resIpTcpConnect(uchar **socketId)**

- Это сообщение содержит запрос **хосту ECI** на открытие соединения между открытым сокетом TCP и одноранговым узлом с использованием протокола сокета.

Определения параметра запроса

socketId: uchar	Сокет (обозначающий номера порта и адрес хоста), из которого устанавливается соединение TCP
peer: Addrinfo	IP-адрес одноранговой узла и IP-порт, для которых должно быть открыто соединение

Определения параметра отклика

socketId: uchar	Идентификатор сокета по **запросу**

Семантическое описание

- Локальный хост предпринимает попытку открыть TCP-соединение между локальным сокетом и одноранговым узлом (IP-адрес, IP-порт).

Предварительные условия

- Сокет открыт для TCP с использованием того же типа IP-адреса (IPv4 или IPv6), что и для peerAddressType.

Постусловия

- Соединение TCP устанавливается или возвращается состояние ошибки.

Коды ошибок, связанных с соединением через TCP и IP, перечисляются в таблице 9.4.4.5.2-1.

Таблица 9.4.4.5.2-1 – Коды ошибок resIpTcpConnect

Название	Описание
ErrIpTcpProtMismatch	См. таблицу 9.4.4.7-1
ErrIpTcpSockNot	
ErrIpTcpIpOffline	
ErrIpTcpConnRefused	
ErrIpTcpConnTimeout	

9.4.4.5.3 Сообщение reqIpTCPSend

C→H reqIpTcpSend(uchar **socketId,** bool **more,** uint **dataLen,** byte **data[])** →
H→C resIpTcpSend(uchar **socketId,** uint **actLen)**

- Это сообщение направляет данные посредством протокола TCP на сокете, подключенном по TCP.

Определения параметра запроса

socketId: uchar	Сокет (обозначающий номер порта и адрес хоста), используемый для отправки данных одноранговому узлу
more: bool	Указывает на то, что данные и предшествующие данные должны быть незамедлительно перенаправлены одноранговому узлу (**more=False**), или на то, что в последующих запросах reqipTcpSend также содержатся дополнительные данные (**more=True**)
dataLen: uint	Объем данных, подлежащих отправке
data: byte[]	Данные, подлежащие отправке

Определения параметра отклика

socketId: uchar	Идентификатор сокета, на котором были сформированы передаваемые данные
actLen: uint	Фактическое количество успешно переданных байтов

Семантическое описание

- Локальный хост направляет **данные** подключенному одноранговому узлу через подключенный сокет TCP/IP с идентификатором **socketID**.

Предварительные условия (запрос)

1) Сокет работает в режиме подключения по TCP/IP.

Постусловия (отклик)

1) Если параметр actLen не равен dataLen, то состояние ошибки сохраняется.

Коды ошибок, связанных с отправкой пакетов TCP, перечисляются в таблице 9.4.4.5.3-1.

Таблица 9.4.4.5.3-1 – Коды ошибок resIpTcpSend

Название	Описание
ErrIpTcpSockNot	См. таблицу 9.4.4.7-1
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.4 Сообщение reqIpTCPRecv

C→H reqIpTcpRecv(uchar **socketId,** uint **maxDataLen)** →
H→C resIpTcpRecv(uchar **socketId,** uint **dataLength,** byte **data[])**

- Это сообщение принимает данные по протоколу TCP на сокете, подключенном по TCP.

Определения параметра запроса

socketId: uchar	Сокет (обозначающий номер порта и адрес хоста), используемый для приема данных одноранговым узлом
maxDataLen: uint	Максимальный объем данных, подлежащих приему

Определения параметра отклика

socketId: uchar	ID сокета, на котором было сформировано принимаемое сообщение
dataLength: uint	Количество байтов данных, полученных от однорангового узла
data: byte[]	Данные, полученные от однорангового узла

Семантическое описание

- Локальный хост принимает **данные** от однорангового узла через сокет, подключенный по TCP/IP, с идентификатором **socketID**.

Предварительные условия (запрос)

1) Сокетом является сокет TCP.

Постусловия (отклик)

1) Все доступные данные длиной до заданного значения возвращаются в **запросе** в поле **maxDataLen**. Если данные недоступны, **отклик** приостанавливается до тех пор, пока соединение не будет закрыто. При этом TCP-соединение будет считаться временно недоступным или локальное подключение к IP-сети будет потеряно.

Коды ошибок, относящиеся к приему пакетов TCP, перечисляются в таблице 9.4.4.5.4-1.

Таблица 9.4.4.5.4-1 – Коды ошибок resIpTcpRecv

Название	Описание
ErrIpTcpSockNot	См. таблицу 9.4.4.7-1
ErrIpTcpIpOffline	
ErrIpTcpClosed	
ErrIpTcpConnTimeout	

9.4.4.5.5 Сообщение reqIpTCPAccept

C→H reqIpTcpAccept(uchar **socketId)** →
H→C resIpTcpAccept(uchar **socketId,** uchar **newSocketId,** Addrinfo **peer)**

- Это сообщение принимает **запрос** на входящее подключение на сокете сервера TCP. Ожидающие **запросы** соединения должны быть обслужены; максимальное количество определяется конфигурацией конкретного **хоста ECI**. Эксплуатационные требования к серверу TCP предлагаются в [b-ITU-T J Suppl. 7].

Определения параметра запроса

socketId: uchar	Сокет (обозначающий номер порта и адрес хоста), используемый для приема **запросов** на соединение

Определения полей сообщения

socketId: uchar	ID сокета, на котором был сформирован запрос
newSocketId: uchar	ID сокета для вновь открытого соединения с одноранговым узлом, который создал **запрос** на подключение. Адрес хоста и порт наследуются от сокета с идентификатором **socketId**
peer: Addrinfo	IP-адрес + IP-порт подключенного однорангового узла

Семантическое описание

- Локальный **хост ECI** ожидает **запросы** на входящее TCP-подключение по IP-адресу/порту, указанному при создании сокета, и открывает новый подключенный сокет, обслуживающий входящий (или ожидающий) **запрос** на подключение. **Отклик** может не последовать, если будет отсутствовать входящий **запрос** или если сокет сервера закрыт.

Предварительные условия (запрос)

1) Сокетом является сокет сервера TCP.

Постусловия (отклик)

1) Новый сокет с открытым TCP/IP-соединением возвращается по **запросу** на любое доступное подключение к сокету сервера или возникает ошибка.

Коды ошибок, относящиеся к приему TCP-подключений, перечисляются в таблице 9.4.4.5.5-1.

Таблица 9.4.4.5.5-1 – Коды ошибок resIpTcpAccept

Название	Описание
ErrIpTcpListSockNot	См. таблицу 9.4.4.7-1
ErrIpTcpNoMoreSockets	

9.4.4.6 API для услуг HTTP(S) GET

9.4.4.6.1 Общие сведения

Хост ECI предоставляет базовые запросы HTTP(S) GET для получения ресурсов от HTTP-сервера на основе IP в интересах клиента. Это позволяет **клиенту ECI** извлекать веб-ресурсы (файлы) из интернет-серверов. HTTPS может, кроме прочего, использоваться для извлечения ресурсов на основе веб-API, в частности импортировать или экспортировать данные, как определяется в пункте 9.7.2 и пункте 7.8.4.2.

Защита обеспечивается посредством протокола HTTPS (TLS) основной конфигурации TLS оборудования **CPE**.

ПРИМЕЧАНИЕ. – Эта система защиты в общем случае не должна использоваться для обеспечения целостности системы защиты контента для **клиентов ECI**, но может быть использована для предотвращения распределенных атак на отказ в обслуживании (DDOS) и других возможных попыток воздействия на **клиентов ECI**.

Хост ECI поддерживает **клиентов ECI** с использованием минимального объема ресурсов для формирования запросов HTTP Get. Значения предлагаются в [b-ITU-T J Suppl. 7].

API-сообщения для HTTP(S) Get передаются в таблице 9.4.4.6.1-1.

Таблица 9.4.4.6.1-1 – Сообщения API HTTP Get

Сообщение	Тип	Направление	Маркер	Описание
reqHttpGetFile	A	C→H	0x0	Выполняет запрос HTTP Get по URL и сохраняет результат в файле
reqHttpGetData	A	C→H	0x1	Выполняет запрос HTTP Get по URL и передает результат в виде данных для клиента

9.4.4.6.2 Применимые спецификации

ПРИМЕЧАНИЕ. – Указанные здесь спецификации не являются необходимой составной частью системы защиты ECI, о которой идет речь в пункте 9.4.4.6.1.

Конфигурация протоколов HTTP и HTTPS для реализации API **клиента ECI** должна соответствовать HTTP1.1 [IETF RFC 7230] и [IETF RFC 7231].

Реализация протокола безопасности транспортного уровня (TLS), используемая для оказания услуг **клиенту ECI** по протоколу HTTP, должна соответствовать требованиям стандарта TLS 1.3 [IETF RFC 8446]. В целях обратной совместимости должен поддерживаться протокол TLS 1.2 с соблюдением ограничений TLS 1.3 и следующих правил:

1) TLS 1.2, см. [IETF RFC 5246].

2) TLS AES-GCM, см. [IETF RFC 5288].

3) TLS-расширениям, см. [IETF RFC 6066].

4) PKIX/X.509 [IETF RFC 5280] + обновления [IETF RFC 6818].

Все используемые конфигурации TLS 1.2 должны поддерживать следующие наборы шифров, как определяется в [IETF RFC 5246]:

1) TLS_RSA_WITH_AES_128_CBC_SHA256.
2) TLS_DHE_RSA_WITH_AES_128_GCM_SHA256.

Могут поддерживаться дополнительные наборы шифров для TLS 1.2 с соблюдением ограничений, установленных для TLS 1.3.

При выборе наборов шифров TLS 1.2 используются следующие правила:

1) Набор TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 должен быть шифром по умолчанию.
2) Наборы шифров AEAD имеют приоритет.
3) Обмен ключами на основе DHE имеет приоритет.
4) Ключи длиной более 128 битов не должны иметь приоритет.
5) Не следует использовать 3DES.
6) RC4 не должен использоваться (как указывается в [W3C PNG]).
7) MD5 не должен использоваться (как указывается в [IETF RFC 6151]).

Применяются следующие правила обработки:

1) TLS 1.2 – минимальная версия, которую требуют все объекты **ECI**.
2) SSL 2.0 и 3.0 не должны использоваться.
3) Не допускается использовать повторное согласование.
4) Не допускается использовать процедуру сжатия (приемлемо с GCM).
5) Простые числа для DH/DHE должны иметь длину не менее 1024 битов и должны проверяться во время квитирования TLS.
6) Проверка **сертификатов** и хостов должна соответствовать требованиям PKIX [IETF RFC 5280] и [IETF RFC 6125].

При выдаче корневых сертификатов, используемых для аутентификации другой стороны в соединении TLS, следует руководствоваться актуальным списком, например https://cabforum.org/browser-os-info/.

Оборудование CPE должно предоставлять **производителю оборудования CPE** возможность удалять корневые сертификаты или отказывать им в доверии после изготовления оборудования. Это может быть осуществлено путем обновления микропрограммного обеспечения, но предпочтительнее реализовать специальный механизм обновления корневых сертификатов, что могло бы обеспечить бóльшую оперативность обновлений. **Производитель оборудования CPE** вправе по своей инициативе удалить из **оборудования CPE** обязательный корневой сертификат или отказать такому сертификату в доверии в ответ на угрозу безопасности. В **оборудовании CPE** должна быть предусмотрена возможность безопасного добавления новых корневых сертификатов в уже изготовленное оборудование для поддержания функциональной совместимости с серверами.

Дополнительные указания по конкретным конфигурациям приведены в правилах обработки, изложенных в материалах CA/Browser Forum [b-CA Browser] и [b-NIST SP 800-52r2].

ПРИМЕЧАНИЕ. –Для обеспечения оперативной совместимости HTTP-серверы, нацеленные на поддержку **клиентов ECI** с услугами на основе HTTP, должны поддерживать совместимые режимы, опции и применимые рекомендации для клиента HTTP, определяемые в настоящем документе.

9.4.4.6.3 Сообщение ReqHttpGetFile и 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[]**)

- Это **сообщение** отправляет **хосту ECI** запрос на выполнение HTTP-запроса для получения файла и возврата статуса HTTP по завершении.
- resHttpGetFile возвращает ресурс в виде файла в файловой системе клиента.

- resHttpGetData возвращает ресурс в виде сообщения с ограниченным размером.

Определения параметра запроса

fname: fileName	Имя файла, в котором **хост ECI** хранит результат запроса (отправленные данные). Любые существующие данные перезаписываются
url: char[]	URL в кодировке UTF-8 [IETF RFC 7230]. Нестандартные номера порта могут быть включены в URL-адрес. TLS должен использоваться для URL-адресов, соответствующих https URI Scheme в [IETF RFC 7230]
userAgent: char[]	Определяет поле заголовка User-Agent для использования в качестве заголовка HTTP. **Клиенты ECI** могут указывать конкретное значение **url**, ожидаемое HTTP-сервером (см. примечание)
redirs: unit	Максимальное количество переадресаций, допустимое для завершения запроса. Минимальные характеристики для **redirs** предлагаются в [b-ITU-T J Suppl. 7]
timeout: unit	Время ожидания в миллисекундах для завершения HTTP-запроса. В случае задержки запрос будет прерван и в **отклике** будет возвращена ошибка в связи с задержкой
ПРИМЕЧАНИЕ. – Не рекомендуется применять User-Agent в качестве механизма контроля доступа или выбора ресурса и следовать целевому использованию, определяемому в [IETF RFC 7231].	

Определения параметра отклика

httpStatus: uint	Значение HTTP-статуса
data: byte[]	Данные результата HTTP GET в сетевом порядке. Максимальный размер ограничивается размером буфера сообщения

Подробная семантика

- **Хост ECI** должен обеспечивать поддержку HTTP-запросами широкого спектра файлов и видов медиаданных. Рекомендуется не включать поле заголовка Accept в заголовок HTTP-запроса. Если заголовок Accept добавлен, для извлечения ресурса должны быть допустимы следующие MIME-типы кодирования контента: application/octet-stream, application/json, image/jpeg, image/png, image/gif, text/plain, text/html, text/css, tex /xml и text/javascript.
- **Хост ECI** должен обеспечивать, чтобы заголовок HTTP-запроса Accept-Encoding передавал информацию о приемлемости следующего типа кодирования контента – gzip.

Постусловия (отклик)

1) Ресурс по **url** был извлечен и сохранен в именах файлов **fname** (для **resHttpGetFile**) либо возвращен в виде данных (для **rerHttpGetData**) или произошла ошибка.

Коды ошибок, связанных с ResHttpGetFile и resHttpGetData, перечислены в таблице 9.4.4.6.3-1.

Таблица 9.4.4.6.3-1 – Коды ошибок resHttpGetFile и resHttpGetData

Название	Описание
ErrHttpGetNoSockets	См. таблицу 9.4.4.6.4-1
ErrHttpGetProtNotAvail	
ErrHttpGetPortNotAvail	
ErrHttpHostUnknown	
ErrHttpDnsOffline	
ErrHttpIpOffline	
ErrHttpTimeout	
ErrHttpGetFSFailure	
ErrHttpGetFSExceeded	
ErrHttpGetTlsAuth	
ErrHttpGetRedir	
ErrHttpGetData	

9.4.4.6.4 Коды ошибок для API HTTP Get

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться с помощью сообщений-**откликов** для данного API, перечисляются в таблице 9.4.4.6.4-1.

Таблица 9.4.4.6.4-1 – Коды ошибок для интерфейсов API HTTP Get

Название	Значение	Описание
ErrHttpGetNoSockets	–257	См. соответствующее значение для кода ошибок в таблице 9.4.4.7-1 для API IP-сокета
ErrHttpGetProtNotAvail	–258	
ErrHttpGetPortNotAvail	–259	
ErrHttpHostUnknown	–261	
ErrHttpDnsOffline	–263	
ErrHttpIpOffline	–267	
ErrHttpTimeout	–270	HTTP-запрос не мог быть завершен в течение заданного в этом запросе времени ожидания
ErrHttpGetFSFailure	–512	Значение +256 соответствует значению кодов ошибок в таблице 9.4.5.5-1 для API файловой системы
ErrHttpGetFSExceeded	–514	
ErrHttpGetTlsAuth	–768	Сервер или данные не могли пройти аутентификацию по протоколу TLS
ErrHttpGetRedir	–784	Превышено количество переадресаций
ErrHttpError	–785	Ресурс не мог быть получен от сервера; причина обозначается кодом ошибки HTTP
ErrHttpGetData	–786	Данные ресурса превысили максимально допустимую длину поля данных

9.4.4.7 Коды ошибок для API IP-сокета

Значения ошибок, связанных с конкретным интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.4.4.7-1.

Таблица 9.4.4.7-1 – Коды ошибок для интерфейсов API IP-сокета

Название	Значение	Описание
ErrIpSourceProt	–256	Неверная комбинация источника и протокола
ErrIpNoSockets	–257	Нет дополнительных доступных сокетов
ErrIpProtNotAvail	–258	Протокол недоступен
ErrIpPortNotAvail	–259	Запрошенный порт недоступен
ErrIpSocketNotOpen	–260	Сокет не открыт
ErrIpHostUnknown	–261	**Хост ECI** неизвестен
ErrIpHost	–262	**Хост ECI** известен, но нет доступных адресов (для IP-адресов заданного типа)
ErrDnsOffline	–263	Услуга DNS отключена от сети (возможно, временно)
ErrIpUdpProtMismatch	–264	Адрес одноранговой узла не соответствует протоколу сокета
ErrIpUdpSockNot	–265	Сокет не является сокетом UDP
ErrIpUdpTooLong	–266	Слишком длинная датаграмма для одного сообщения UDP
ErrIpUdpIpOffline	–267	IP-соединение отключено от сети (нет связи с одноранговым узлом)
ErrIpTcpProtMismatch	–268	Адрес одноранговой узла не соответствует протоколу сокета
ErrIpTcpSockNot	–269	Сокет не является сокетом TCP
ErrIpTcpIpOffline	–258	Локальное IP-подключение к интернету в данный момент отсутствует
ErrIpTcpConnRefused	–259	Соединение через данный порт не принимается хостом одноранговой узла
ErrIpTcpConnTimeout	–260	Невозможно получить **отклик** от **хоста ECI** одноранговой узла
ErrIpTcpClosed	–261	TCP-подключение отсутствует или более недоступно
ErrIpTcpListSockNot	–262	Сокет не является сокетом сервера TCP
ErrIpTcpNoMoreSockets	–263	**Запрос** входящего соединения получен, однако сокеты хоста отсутствуют
RFU	Прочее	Зарезервировано для использования в будущем

9.4.5 Интерфейс API для доступа к файловой системе

9.4.5.1 Введение

Клиент ECI имеет доступ к частной файловой системе для хранения ограниченного количества данных, которые должны сохраняться в течение жизненных циклов **клиента ECI**, циклов включения оборудования **CPE**, при системных сбоях и т. д. в условиях нормальной эксплуатации. Надежность, как минимум, должна соответствовать надежности обычной файловой системы оборудования **CPE**, то есть сбои могут происходить только при некоторых исключительных обстоятельствах, что может привести к неудобствам для **пользователя**. Система безопасности, управляющая **клиентом ECI**, должна предотвращать неоправданную потерю прав доступа **пользователя** к контенту. Файловая система не защищена. Должна быть исключена возможность вмешательства любых объектов, кроме заданного **клиента ECI** и поддерживающего его **хоста ECI** в нормальных условиях (то есть с оборудованием **CPE** и **хостом ECI**, работающими без риска сбоев).

Абстракция файловой системы не предполагает наличия подкаталогов. Доступны основные службы каталога. Функции доступа к файловой системе аналогичны запросам для файловых систем Unix/Linux/Posix, таким как open, close, write, read, lseek, opendir, readdir и lstat.

Минимальный объем хранилища файловой системы должен быть доступен для каждого **клиента ECI**, если он сохраняется **пользователем**. Этот объем предлагается в [b-ITUT J Suppl. 7].

Интерфейс API файловой системы делится на три подраздела:

1) Открытие и закрытие файлов.

2) Файл чтения и записи, произвольный доступ и удаление выбранных данных из файла.

3) Службы каталога.

Имена файлов состоят из 8-битовой последовательности символов ASCII, не менее 1 и не более 8, из следующего набора символов (разделенных запятой): A–Z, a–z, 0–9, _ и **завершается** символом NULL. Определение имен файлов (filename) дается в таблице 9.4.5.1-1.

Таблица 9.4.5.1-1 – Структура FileName

```
typedef char fileName[9];
```

Функциональные возможности файлов системных журналов позволяют **клиентам ECI** записывать ограниченные объемы данных с буферизацией, то есть без остановки выполнения операций. Количество файлов системных журналов для каждого **клиента ECI** определяется в xxx (минимум 2 на каждого клиента). Таким образом, эти файлы подходят для ведения журналов, отслеживания и послеаварийного анализа на уровне приложений.

9.4.5.2 Открытие и закрытие файлов

9.4.5.2.1 Общие сведения

Клиенты ECI могут открывать файл для чтения и/или записи, получая объект fileHandle, через который затем может осуществляться доступ для чтения и записи. Если файл не существует, он может быть создан. Конкретный файл обладает свойством file location (расположение файла), указывает на текущее расположение для доступа к файлу.

Файлы FileHandle управляются **хостом ECI**. Закрытый дескриптор файла не должен повторно использоваться сразу после закрытия. Это гарантирует, что несинхронные попытки доступа к файлу со стороны **клиента ECI** не приведут к тому, что доступ будет открыт к неверному файлу.

Сообщения об открытии и закрытии файлов описываются в таблице 9.4.5.2.1-1.

Таблица 9.4.5.2.1-1 – Сообщения об открытии и закрытии файлов

Сообщение	Тип	Направление	Маркер	Описание
reqFileOpen	A	C→H	0x0	Открывает частный файл **клиента ECI**
reqFileClose	A	C→H	0x1	Закрывает открытый файл

9.4.5.2.2 Сообщение reqFileOpen

C→H reqFileOpen(fileName **fname,** uint **fileOpenOptions**) →
H→C resFileOpen(uchar **fileHandle)**

- Это сообщение позволяет **клиенту ECI** посылать **хосту ECI** запрос на открытие файла с определенными разрешениями для доступа.

Определения параметра запроса

fname: filename	Имя файла, подлежащего открытию
fileOpenOptions: unit	Режим доступа, в котором следует открыть файл. Допустимые значения и их содержание определяются в таблице 9.4.5.2.2-1.

Таблица 9.4.5.2.2-1 – Необязательные параметры открытия файла

Название	Биты	Значение	Описание
FileRead	0,1	0b00	Файл открывается для чтения. Расположение файла задается в начале файла
FileWriteAppend	0,1	0b01	Файл открывается для записи; последующие записи добавляются к существующему файлу. Расположение файла задается в конце файла
FileWriteOver	0,1	b11	Файл открывается для записи в любом расположении. Расположение файла задается в конце файла
Not in use	0,1	0b10	Не допускается
LogFileNo	2	0b0	Обычный файл
LogFileYes	2	0b1	Специальный файл журнала, в котором разрешаются синхронные записи
Bits32-2		Прочее	Зарезервировано для использования в будущем

Определения параметра отклика

fileHandle: uchar	Ссылка (дескриптор) на открытый файл

Постусловия (запрос)

1) Файл открывается в требуемом режиме доступа или будет возвращена ошибка. Коды ошибок перечисляются в таблице 9.4.5.2.2-2.

Таблица 9.4.5.2.2-2 – Коды ошибок resfileOpen

Название	Описание
ErrFileNameNotExist	См. таблицу 9.4.5.5-1
ErrFileQuotaExceeded	
ErrFileSystemFailure	

9.4.5.2.3 Сообщение reqFileClose

C→H reqFileClose(uchar **fileHandle) →**
H→C resFileClose()

- Это сообщение закрывает доступ к файлу, открытому с помощью **fileHandle**. Коды ошибок, связанных с закрытием файла, перечисляются в таблице 9.4.5.2.3-1.

Определения параметра запроса

fileHandle: uchar	Дескриптор файла, подлежащего закрытию

Предварительные условия (запрос)

1) fileHandle находится в открытом состоянии.

Постусловия (запрос)

1) Последующие попытки доступа к fileHandle завершатся ошибкой с ErrFileNotOpen.
2) Все ожидающие записи будут зафиксированы (если не произойдет ошибка).

Таблица 9.4.5.2.3-1 – Коды ошибок resfileClose

Название	Описание
ErrFileHandleNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.3 Доступ к файлу

9.4.5.3.1 Общие сведения

Сообщения доступа к файлу позволяют считывать и записывать данные в файле, доступ к которому осуществляется через дескриптор файла, и перепозиционировать текущее расположение в файле для чтения/записи. Определенные примитивы имеют непосредственное отношение к соглашениям Linux/Unix. Определяемые сообщения перечисляются в таблице 9.4.5.3.1-1.

ПРИМЕЧАНИЕ. – Файлы reqFileWrite и reqFileRead имеют большое сходство с файлами reqTcpSend и reqTcpRecv.

Таблица 9.4.5.3.1-1 – Сообщения доступа к файлам

Сообщение	Тип	Направление	Маркер	Описание
reqFileWrite	A	C→H	0x2	Записывает последовательные байты, начиная с текущего расположения файла
reqFileRead	A	C→H	0x3	Считывает последовательные байты, начиная с текущего расположения файла
reqFileSeek	A	C→H	0x4	Перепозиционирует текущее расположение файла
reqFileRemoveData	A	C→H	0x5	Удаляет данные из файла в текущем расположении
callFileDataLog	S	C→H	0x6	Добавляет данные в конце буферизованного файла

9.4.5.3.2 Сообщение reqFileWrite

C→H reqFileWrite(uchar **fileHandle,** bool **sync,** uint **dataLen,** byte **data[]) → H→C resFileWrite(**uchar **fileHandle)**

- Это сообщение записывает байты dataLen в файл, начиная с текущего расположения файла.

Определения параметра запроса

fileHandle: uchar	Дескриптор файла для записи
sync: булева переменная	Если значение равно True, **запрос** записи гарантирует, что состояние файловой системы обновляется в соответствии с этой и всеми предыдущими записями. Если значение равно False, **хост ECI** может буферизовать **запросы** записи (которые могут быть потеряны при сбое системы)
dataLen: uint	Количество байтов для записи в файл
data: byte[]	Данные для записи в файл

Определения параметра отклика

fileHandle: uchar	Дескриптор файла, в который была произведена запись

Предварительные условия (запрос)

1) Файл открыт в режиме записи (режимы FileWriteOver или FileWriteAppend).
2) Запись расположения файла: если файл открыт в режиме FileWriteAppend, расположение файла должно быть записано в конце.
3) Объем записываемых данных не вызывает проблем с квотированием файловой системы.

Постусловия (запрос)

1) Состояние файла будет обновлено, и запись расположения файла будет расширена (в процессе ожидания других буферизованных операций с файлом) с прибавлением dataLen, если не произойдет ошибка.
2) В случае успешной записи и **синхронизации** данные фиксируются в неизменяемом состоянии в файловой системе **хоста ECI**.

Коды ошибок перечисляются в таблице 9.4.5.3.2-1.

Таблица 9.4.5.3.2-1 – Коды ошибок resFileWrite

Название	Описание
ErrFileHandleNotExist	См. таблицу 9.4.5.5-1
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.3 Сообщение reqFileRead

C→H reqFileRead(uchar **fileHandle**, uint **dataLen) →**
H→C resFileRead(uchar **fileHandle**, uint **dataRead,** byte **data[])**

- Это сообщение считывает максимальное количество байтов dataLen из файла, начиная с текущего расположения файла. Коды ошибок, связанных со считыванием данных из файла, перечисляются в таблице 9.4.5.3.3-1.

Определения параметра запроса

fileHandle: uchar	Дескриптор файла для чтения
dataLen: uint	Максимальное количество байтов для чтения

Определения параметра отклика

fileHandle: uchar	Дескриптор файла, который был прочитан
dataRead: uint	Количество байтов, которые были прочитаны и сохранены в **данных**
data: byte []	Считанные данные

Предварительные условия (запрос)

1) Файл открыт.

Постусловия (запрос)

1) Произошла ошибка; или
2) минимальное количество байтов dataLen или оставшихся байтов в файле из последнего расположения считывается из файла; и
3) к записи расположения файла добавлен параметр **dataRead**;
4) если не возникает ошибки, запись расположение файла расширяется параметром **dataLen** или перемещается в конец файла.

Таблица 9.4.5.3.3-1 – Коды ошибок resFileReade

Название	Описание
ErrFileHandleNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.3.4 Сообщение reqFileSeek

C→H reqFileSeek(uchar **fileHandle,** int **offset,** uchar **seekPos) →**
H→C resFileSeek(uchar **fileHandle,** int **remOffset)**

- Это сообщение перемещает указатель в определенную позицию в пределах открытого файла и возвращает части содержимого файла.

Определения параметра запроса

fileHandle: uchar	Дескриптор файла, расположение которого подлежит изменению
offset: int	Смещение от исходного расположения поиска, как определяется значением **seekPos**, которое принимается в качестве расположения файла
seekPos: uchar	См. таблицу 9.4.5.3.4-1

Таблица 9.4.5.3.4-1 – Исходное расположение для поиска файлов

Название	Значение	Описание
FileSeekSet	0x00	Исходное расположение файла находится в начале файла
FileSeekCur	0x01	Исходное расположение файла соответствует текущему расположению файла
FileSeekEnd	0x02	Исходное расположение файла находится в конце файла
RFU	Прочее	Зарезервировано для использования в будущем

Определения параметра отклика

fileHandle: uchar	Дескриптор файла, расположение которого было изменено.
remOffset: int	Разница между указанным смещением и смещением, для которого устанавливается расположение файла

Подробная семантика

- Расположение файла перепозиционируется и определяется в описании параметра **запроса**. Расположение файла не будет перемещаться за пределы конца файла или размещаться перед началом файла. Разница между запрашиваемым смещением и фактическим смещением от исходного расположения файла возвращается в виде результирующего параметра **remOffset**. Коды ошибок перечисляются в таблице 9.4.5.3.4-2.

Предварительные условия (запрос)

1) Файл открыт.

Постусловия (запрос)

1) Произошла ошибка; или
2) расположение файла задается как указано выше; и
3) **remOffset** отражает разницу между смещением и определяемым выше фактическим расположением файла.

Таблица 9.4.5.3.4-2 – Коды ошибок resFileReade

Название	Описание
ErrFileHandleNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.3.5 Сообщение reqFileRemoveData

C→H reqFileRemoveData(uchar **fileHandle,** bool **sync,** uint **dataLen) →**
H→C resFileRemoveData(uchar **fileHandle)**

- Это сообщение удаляет байты dataLen из файла, начиная с текущего расположения файла.

Определения параметра запроса

fileHandle: uchar	Дескриптор файла
sync: булева переменная	Если значение равно True, **запрос** записи гарантирует, что состояние файловой системы обновляется в соответствии с этой и всеми предыдущими записями. Если значение равно False, **хост ECI** может буферизовать **запросы** записи (которые могут быть потеряны при сбое системы)
dataLen: uint	Количество байтов для удаления из файла. Если это количество превышает длину файла, то удаляются байты только до конца файла

Определения параметра отклика

fileHandle: uchar	Дескриптор файла, в который была произведена запись

Предварительные условия (запрос)

1) Файл открыт в режиме записи (режим FileWriteOver).

Постусловия (запрос)

1) Состояние файла подлежит обновлению. Расположение файла остается прежним.

2) В случае успешного удаления и **синхронизации** данные фиксируются в неизменяемом состоянии в файловой системе **хоста ECI.**

Коды ошибок определяются в таблице 9.4.5.3.5-1.

Таблица 9.4.5.3.5-1 – Коды ошибок resFileWrite

Название	Описание
ErrFileHandleNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	
ErrFileWriteNot	

9.4.5.3.6 Сообщение callFileDataLog

C→H callFileDataLog(uchar **fileHandle,** uint **dataLen**, byte **data[])**

- Это сообщение добавляет байты dataLen (в данных) в конце файла, используя системный буфер.

Определения параметра вызова

fileHandle: uchar	Дескриптор файла
dataLen: uint	Количество байтов для добавления в файл журнала
data[]: byte	Данные для записи

Предварительные условия (вызов)

1) Файл открыт в режиме записи (режим FileWriteOver или FileWriteAppend).
2) Расположение файла задается в конце файла.
3) Объем записываемых данных не вызывает проблем с квотированием файловой системы.

Постусловия (вызов)

1) Состояние файла обновляется, и запись расположения файла расширяется с прибавлением dataLen, если не произойдет ошибка.
2) Результат будет зафиксирован в файловой системе хоста ECI, если не произойдет сбой системы.

Подробная семантика

1) Хост ECI должен буферизовать данные и добавлять их в конец файла по мере целесообразности.
2) Максимальное пространство буфера, предусмотренное для журнала с этой целью, предлагается в [b-ITU-T J Suppl. 7].

Коды ошибок определяются в таблице 9.4.5.3.6-1.

Таблица 9.4.5.3.6-1 – Коды ошибок resFileLog

Название	Описание
ErrFileHandleNotExist	Определение см. в таблице 9.4.5.5-1
ErrFileQuotaExceeded	
ErrFileSystemFailure	
ErrFileWriteNot	
ErrFileLogNot	

9.4.5.4 Службы каталога

9.4.5.4.1 Общие сведения

Службы каталога предлагают функции сканирования доступных файлов **клиента ECI**. Для каждого файла характерно его уникальное имя, а также такие атрибуты, как размер и время последнего изменения. Перечень доступных сообщений приводится в таблице 9.4.5.4.1-1.

ПРИМЕЧАНИЕ. – Атрибут времени имеет ту же степень целостности, что и файловая система и само содержание файла.

Таблица 9.4.5.4.1-1 – Сообщения службы каталога файлов

Сообщение	Тип	Направление	Маркер	Описание
reqFileStat	A	C→H	0x07	Возврат размера и времени изменения файла
reqFileCreate	A	C→H	0x08	Создание нового файла
reqFileDelete	A	C→H	0x09	Удаление файла
reqFileDir	A	C→H	0x0A	Перечисляет имена файлов, доступных в файловой системе **клиентов ECI**

9.4.5.4.2 Сообщение reqFileStat

C→H reqFileStat(fileName **filename) →**
H→C resFileStat(uint **size;** long **mtime)**

- Это сообщение позволяет **клиенту ECI** направлять запрос **хосту ECI**, чтобы получить размер файла и время последнего изменения сохраняемого файла.

Определения параметра запроса

filename: filename	Имя файла, для которого должны быть получены свойства

Определения параметра отклика

size: uint	Размер файла (в байтах)
mtime: long	Точное время последнего изменения синхронизированного файла

Предварительные условия (запрос)

1) Имя файла обозначает существующий файл в файловой системе.

Постусловия (запрос)

1) **size** и **mtime** отражают свойства файла с именем **filename** или произошла ошибка.

Коды ошибок перечисляются в таблице 9.4.5.4.2-1.

Таблица 9.4.5.4.2-1 – Коды ошибок resFileStat

Название	Описание
ErrFileNameNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.4.3 Сообщение reqFileCreate

C→H reqFileCreate(fileName **filename) →**
H→C resFileCreate()

- Это сообщение позволяет **клиенту ECI** направлять запрос **хосту ECI**, чтобы создать новый пустой файл. Любой существующий файл с тем же именем удаляется.

Определения параметра запроса

filename: filename	Имя нового пустого файла, который должен быть создан

Подробная семантика

- Создаваемый файл должен существовать после сбоя системы, если только файловая система не была повреждена.

Постусловия (запрос)

1) Пустой файл с именем filename существует в файловой системе **клиента ECI** с отметкой о времени внесения изменений, соответствующем текущему моменту, или произошла ошибка.

Коды ошибок перечисляются в таблице 9.4.5.4.3-1.

Таблица 9.4.5.4.3-1 – Коды ошибок resFileCreate

Название	Описание
ErrFileQuotaExceeded	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.4.4 Сообщение reqFileDelete

C→H reqFileDelete(fileName **filename) →**
H→C resFileDelete()

- This Message deletes a file with name **filename**.

Определения параметра запроса

filename: fileName	Имя нового пустого файла, который должен быть создан

Подробная семантика

- Удаляемый файл не должен существовать после сбоя системы, если только файловая система не была повреждена.

Постусловия (запрос)

1) Файл с именем **filename** не существует в файловой системе.

Коды ошибок перечисляются в таблице 9.4.5.4.4-1.

Таблица 9.4.5.4.4-1 – Коды ошибок resFileDelete

Название	Описание
ErrFileNameNotExist	См. таблицу 9.4.5.5-1
ErrFileSystemFailure	

9.4.5.4.5 Сообщение reqFileDir

C→H reqFileDir(ushort **maxNr) →**
H→C resFileDir(uint **listLen;** fileName **dirList[])**

- Это сообщение содержит список позиций max. maxNr в составе имен файла. Порядок списка не определен.

Определения параметра запроса

maxNr: ushort	Максимальное количество имен файлов, которые будут извлечены

Определения параметра отклика

listLen: uint	Длина списка в байтах
dirList: filename []	Массив имен файлов, доступных для **клиента ECI**

Коды ошибок перечисляются в таблице 9.4.5.4.5-1.

Таблица 9.4.5.4.5-1 – Коды ошибок resFileDelete

Название	Описание
ErrFileSystemFailure	См. таблицу 9.4.4.7-1

9.4.5.5 Коды ошибок для API файловой системы

Значения ошибок, связанных с конкретным интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.4.5.5-1.

Таблица 9.4.5.5-1 – Коды ошибокAPI файловой системы

Название	Значение	Описание
ErrFileSystemFailuret	–256	Поврежденная или демонтированная файловая система
ErrFileNameNotExist	–257	Имя файла не существует в файловой системе
ErrFileQuotaExceeded	–258	Ресурсы файловой системы для **клиента ECI** превышены
ErrFileNameNotExists	–259	Имя файла не существует в файловой системе **клиента ECI**
ErrFileHandleNotExists	–260	Дескриптор файла не существует (вероятно, был закрыт ранее)
ErrFileAppendNot	–261	Попытка записи в файл произошла не в конце файла
RFU	Прочее	Зарезервировано для использования в будущем

9.4.6 Интерфейс API для доступа к ресурсу времени/часов

9.4.6.1 Введение

Клиент ECI получает доступ к событиям таймера и времени суток через простой API.

Устойчивость работы часов должна задаваться режимом обеспечения устойчивости, подходящим для всех приложений в **экосистеме ECI**.

- Если требуется, чтобы **экосистема ECI** поддерживала отмену отката системы хранения файлов или зависимое от времени выражение прав в отсутствие доступа к сети, часы должны быть устойчивыми, чтобы операции над локальной системой хранения с меткой времени, проставленной по этим часам, были в достаточной мере защищены от манипуляций.

ПРИМЕЧАНИЕ. – Комбинируя часы и API таймера, можно создавать регулярные события таймера.

Интерфейсы API таймера и часов делятся на две категории:

1) API таймера;

2) API часов.

9.4.6.2 API таймера

9.4.6.2.1 Общие сведения

API таймера позволяет **клиенту ECI** устанавливать таймер, который отправляет **отклик** в заданное время. При необходимости данное событие может быть отменено **клиентом ECI**. Количество одновременно ожидающих обработки таймеров может быть ограничено для конкретной конфигурации. Минимальное количество ожидающих обработки таймеров, которые поддерживает **хост ECI** для каждого **клиента ECI**, предложено в [b-ITU-T J Suppl. 7]. Сообщения для API таймера определяются в таблице 9.4.6.2.1-1.

Таблица 9.4.6.2.1-1 – Сообщения API таймера

Сообщение	Тип	Направление	Маркер	Описание
reqTimerEvent	A	C→H	0x0	Устанавливает событие таймера в будущем
reqTimerCancel	A	C→H	0x1	Отменяет ранее установленное событие таймера

9.4.6.2.2 Сообщение reqTimerEvent

C→H reqTimerEvent(uint **timeInterval) →**
H→C resTimerEvent()

- Это сообщение устанавливает таймер в будущем и принимает **отклик** по истечении установленного в таймере времени.

Определения параметра запроса

timeInterval: uint	Время в миллисекундах в будущем

Постусловие (запрос)

- Если сообщение reqTimerCancel не принимается первым, то по истечении интервала timeInterval в миллисекундах **клиенту ECI** посылается сообщение resTimerEvent.

Предварительные условия (отклик)

- Время таймера истекло, и сообщение reqTimerCancel для таймера не принято.

Коды ошибок перечисляются в таблице 9.4.6.2.2-1.

Таблица 9.4.6.2.2-1 – Коды ошибок resTimerEvent

Название	Описание
ErrTimerMaxExceeded	См. таблицу 9.4.6.4-1

9.4.6.2.3 Сообщение reqTimerCancel

C→H reqTimerCancel(msgId **id) →**
H→C resTimerCancel()

- Это сообщение отменяет ранее установленный таймер для идентификатора сообщений исходного **запроса**.

Определения параметра запроса

id: msgId	Отмена таймера, который был установлен посредством асинхронного сообщения с идентификатором message id.

Предварительные условия (запрос)

1) Идентификатор возвращен как результат события reqTimerEvent, при этом время таймера еще не истекло.

Постусловия (отклик)

1) Таймер отменяется: сообщение resTimerCancel не отправляется или возвращается ошибка.
2) Если таймер отменен, но событие **resTimerEvent** принято до сообщения **resTimerCancel**, возникает ошибка TimerExpired.

9.4.6.3 API часов

9.4.6.3.1 Общие сведения

API часов позволяет **клиенту ECI** считывать показания часов в виде целого числа и преобразовывать его в показания местного времени. Сообщения API часов перечисляются в таблице 9.4.6.3.1-1.

Таблица 9.4.6.3.1-1 – Сообщения API часов

Сообщение	Тип	Направление	Маркер	Описание
getTime	S	C→H	0x3	Считывает показания локальных системных часов в виде целого числа
callLocaltime	S	C→H	0x4	Преобразует целочисленное значение времени в локальное время (localtime)

9.4.6.3.2 Сообщение getTime

C→H long **getTime()**

- Это сообщение возвращает время в секундах с 1 января 1970 года, 0:00 GMT.

9.4.6.3.3 Сообщение callLocaltime

C→H callLocaltime(long **time;** tm ***tim)**

- Это сообщение преобразует время **time** в представление мирового времени и определяется в структуре **tim**. Аналогично функции localtime библиотеки c-library из <time.h>.

Определения параметра вызова

time: long	Время в целочисленном представлении секунд с 1 января 1970 года, 0: 00 GMT для преобразования в местное время
tim: tm *	Указатель для структуры tm, который устанавливается на местное время. Структура **tm** определяется в таблице 9.4.6.3.3-1

Таблица 9.4.6.3.3-1 – Определение типа для структуры представления мирового времени tm

```
typedef struct tm {
    int tm_sec;   // 0 .. 59 (секунды) или 60 при наличии дополнительной секунды
    int tm_min;   // 0 .. 59 (минуты)
    int tm_hour       // 0 .. 23 (часы)
    int tm_mday;  // 1 .. 31 (число месяца)
    int tm_mon;       // 1 .. 12 (месяц)
    int tm_year;  // year – 1900
    int tm_wday;  // 0 .. 6 (день недели; 0 = воскресенье)
    int tm_yday;  // 0 .. 365 (день в году, 0= 1 января)
    int tm_isdst; // 1 = переход на летнее время, 0=нет перехода на летнее время
    char tm_zone[15]; // строка для часового пояса, например, GMT, CET
    int tm_gmtoff;// сдвиг местного времени относительноGMT
} tm ;
```

9.4.6.4 Коды ошибок для API времени и часов

Значения ошибок, связанных с конкретным интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.4.6.4-1.

Таблица 9.4.6.4-1 – Коды ошибок API времени и часов

Название	Значение	Описание
ErrTimerMaxExceeded	256	Превышено максимальное время таймера
RFU	Прочее	Зарезервировано для использования в будущем

9.4.7 API для доступа к системе управления энергопотреблением

9.4.7.1 Введение

Клиент ECI имеет доступ к интерфейсу управления энергопотреблением **хоста ECI**. Этот интерфейс разрешает **клиенту ECI** переходить в режим пониженного энергопотребления, в том числе в результате перехода системы в режим ожидания, а также позже перезапускать оборудование **CPE** и **клиента ECI** из режима ожидания для выполнения фоновых операций. **Хост ECI** имеет следующие режимы энергопотребления:

- **PwrOn – Хост ECI** функционирует и не собирается переходить в режим пониженного энергопотребления;
- **PwrToStby – Хост ECI** намеревается переходить в режим ожидания (с возможностью возврата в режим "включен постоянно"). Все **клиенты ECI**, как правило, получают запрос на переход в режим пониженного энергопотребления;
- режим ожидания – Хост ECI и клиент ECI не функционируют. Оборудование CPE (а следовательно, хост ECI и клиент ECI) может выходить из этого состояния по заранее условленному событию (как правило, таймеру);
- Power-off – электропитание оборудования CPE отключено. Хост ECI и клиент ECI не функционируют.

Клиенты ECI могут работать в обычном режиме энергопотребления и могут просто отключаться в тот момент, когда **хост ECI** сочтет это необходимым. Помимо этого, **клиенты ECI** могут запрашивать переход в управляемый режим путем отправки сообщения **reqPwrInfo(PwrInfoOn)**. В этом режиме они уведомляются о намерении **хостов ECI** перейти в режим пониженного энергопотребления посредством сообщения **reqPwrChange**, которое **клиент ECI** может подтвердить, отправив сообщение **resPwrChange(PwrDown)**, или отложить, указав соответствующее значение параметра в сообщении **resPwrChange(PwrUp)**, до завершения операций и готовности к переходу в режим ожидания. **Хост ECI** регулярно повторяет отправку сообщения **reqPwrChange**.

ПРИМЕЧАНИЕ. – Невозможно полностью гарантировать способность **клиента ECI** завершить все операции (например, в случае неконтролируемых перебоев в электропитании или продолжительной задержки готовности к переходу в режим ожидания).

На рисунке 9.4.7.1-1 представлено состояние **хоста ECI**, а также условия перехода в другое состояние и действия/сообщения, которые передаются при смене состояния **клиентам ECI**, работающим в управляемом режиме.

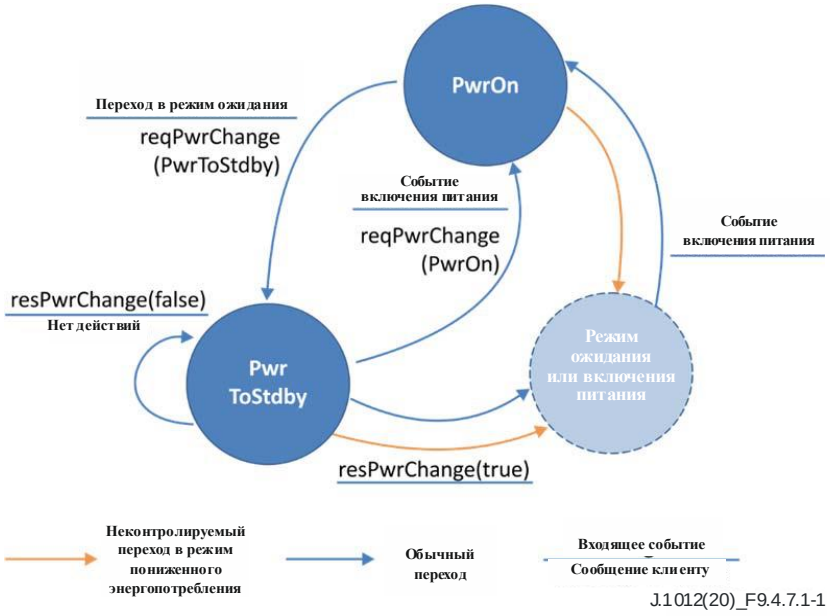


Рисунок 9.4.7.1-1 – Режимы энергопотребления хоста ECI и основные виды взаимодействия с клиентом, работающим в управляемом режиме

Клиенты ECI и **хосты ECI** должны быть способны восстанавливать систему после *неконтролируемого перехода на пониженное энергопотребление*. В подобных случаях допускается частичная временная блокировка стандартного набора функций **клиента ECI** и **хоста ECI**, позволяющая свести к минимуму проблемы, которые могут возникнуть у **пользователя**.

Устройства **CPE** могут обладать функциями запуска (то есть перехода в обычный режим питания) из режима пониженного энергопотребления, связанного с сетевыми событиями, а также другими энергосберегающими режимами. Интерфейсом **ECI** не определен конкретный порядок действий в таких режимах энергопотребления. Взаимодействие с **хостом ECI** или **клиентами ECI**, не связанное с услугами **хоста ECI** и **клиента ECI**, должно сохраняться независимо от того, находится ли **хост ECI** в состоянии **PwrOn** или **PwrToStdby**. Другими словами, не существует никакого конкретного состояния в случае приостановки выполнения операций.

Клиенты ECI должны иметь возможность направлять **хосту ECI** запрос на выход из режима ожидания в определенный момент времени в будущем и отправлять сообщение **клиенту ECI**.

Интерфейс API управления энергопотреблением разделяется на следующие группы сообщений:

1) Переключение режимов энергопотребления: управление упорядоченным отключением **клиентов ECI**. Подробная информация приводится в пункте 9.4.7.2.

2) Функции регулируемого во времени выхода из режима ожидания для **клиентов ECI**. Подробная информация приводится в пункте 9.4.7.3.

9.4.7.2 Определение сообщений API переключения режимов энергопотребления

9.4.7.2.1 Общие сведения

В этом пункте, касающемся API управления энергопотреблением, определяются функциональные возможности, позволяющие **клиентам ECI** сообщать об отключении электропитания при наступлении объявленного события пониженного энергопотребления в **хосте ECI** в целях оптимизации оказания услуг **пользователю**. Определяемые сообщения перечисляются в таблице 9.4.7.2.1-1.

Таблица 9.4.7.2.1-1 – Сообщения о переключении режимов энергопотребления

Сообщение	Тип	Направление	Маркер	Описание
getPwrStatus	S	C→H	0x0	Получает сообщение о текущем состоянии электропитания
setPwrInfo	S	C→H	0x1	Запрашивает оповещение об изменении состояния электропитания
reqPwrChange	A	H→C	0x2	Оповещение об изменении состояния электропитания

Клиенты ECI не должны прекращать работу после отправки сообщения **resPwrInfo(PwrDown)**, однако должны быть готовы возобновить работу в обычном режиме при получении сообщения **reqPwrChange(PwrOn)**.

9.4.7.2.2 Сообщение getPwrStatus

C→H uchar **getPwrStatus()**

- Это сообщение возвращает текущее состояние электропитания **хоста ECI**.

Определение свойства: см. таблицу 9.4.7.2.2-1.

Таблица 9.4.7.2.2-1 – Значения состояния электропитания хоста

Название	Значение	Описание
PwrOn	0x00	IP-адрес **хоста ECI** по умолчанию
PwrToStdby	0x01	IP-адрес **хоста ECI,** используемый для (интернет) связи по сети WAN
RFU	Прочее	Зарезервировано для использования в будущем

9.4.7.2.3 Сообщение setPwrInfo

C→H setPwrInfo(bool **pwrInfo)**

- Это сообщение позволяет входить в режим управляемого пониженного электропитания и выходить из него, а также управлять отправкой **хостом ECI** для **клиента ECI** сообщений **resPwrChange** при изменениях режима энергопотребления.

Определение свойства

- **pwrInfo**, равное **true**, соответствует управляемому режиму энергопотребления; **pwrInfo**, равное **false**, соответствует неуправляемому режиму энергопотребления.

Семантическое описание

- Если **pwrInfo** равно **True**, **хост ECI** оповещает **клиента ECI** об изменениях режима энергопотребления и не переводит **клиента ECI** в режим пониженного питания, пока **клиент ECI** не подтвердит сообщение reqPwrChange(PwrToStby). Если **pwrInfo** равно **False**, **хост ECI** не оповещает **клиента ECI** об изменениях режима энергопотребления и переводит **клиента ECI** в режим пониженного питания по своему усмотрению.
- После запуска состояние **PowerInfo** для каждого **клиента ECI** равно **False**.

ПРИМЕЧАНИЕ. – **Клиентам ECI**, функционирование которых зависит от управляемого пониженного электропитания, не рекомендуется приступать к выполнению операций, для которых критичен цикл понижения электропитания, до отправки **хосту ECI** сообщения **reqPwrInfo(True**.

9.4.7.2.4 Сообщение reqPwrChange

H → C reqPwrChange(uchar **hostPwrState)** →
C→H resPowerChange(bool **ready)**

- Это **сообщение** передает информацию об изменении состояния электропитания; если аргумент равен **PwrToStdby**, то **клиент ECI** может либо дать подтверждение и перейти в режим ожидания контролируемым способом, либо отклонить запрос, если в данный момент им выполняются важные программные задачи.

Определения параметра запроса

hostPwrState: uchar	Новое состояние электропитания **хоста ECI**. Возможные значения определяются в таблице 9.4.7.2.2-1

Определения параметра отклика

ready: булева переменная	Указывает на готовность **клиента ECI** к переходу в режим ожидания

Семантическое описание

- **Хост ECI** повторно передает это сообщение в том случае, если отклик **клиента ECI** отрицательный (не готов). Значения минимальной частоты повторения и времени ожидания предлагаются в [b-ITU-T J Suppl. 7].

Предварительные условия (запрос)

1) PwrInfo == True.
2) Произошло (последнее) изменение состояния электропитания в **хосте ECI**, а **клиент ECI** еще не подтвердил готовность к переходу в режим ожидания.

Постусловия (отклик)

1) **Клиент ECI** готов к переходу в режим ожидания, если **ready** == **True**, и не готов, если **ready == False.**

Коды ошибок определяются в таблице 9.4.7.2.4-1.

Таблица 9.4.7.2.4-1 – Коды ошибок ansPwrChange

Название	Описание
ErrPwrInfoNot	См. таблицу 9.4.7.4-1

ПРИМЕЧАНИЕ. – **Хосты ECI** принимают **ошибку ErrPwrInfoNot** только в целях информации.

9.4.7.3 Определение сообщений о выходе из режима ожидания

9.4.7.3.1 Общие сведения

В этом пункте, касающемся API управления энергопотреблением, определяются функциональные возможности, позволяющие **клиентам ECI** возобновлять функционирование в предварительно запрограммированное время, выводя оборудование **CPE** из режима ожидания, если это необходимо. Определяемые сообщения перечисляются в таблице 9.4.7.3-1.

Таблица 9.4.7.3-1 – Сообщения для выхода из режима ожидания

Сообщение	Тип	Направление	Маркер	Описание
setPwrWakeup	set	C→H	0x3	Устанавливает время выхода из режима ожидания для **клиента ECI**
reqPwrWakeupEvent	A	H→C	0x4	Сигнализирует об истечении времени таймера для выхода из режима ожидания

9.4.7.3.2 Сообщение setPwrWakeup

C→H setPwrWakeup(uint **time)**

- Это сообщение для установки таймера. По истечении **времени хост ECI** выводит **клиента ECI** из режима ожидания, если это необходимо, и отправляет сообщение а **reqPwrWakeupEvent()**.

Определение свойства

time: uint	Время в секундах до момента генерации **хостом ECI** события вывода из режима ожидания для **клиента ECI**. Если значение равно 0, то **клиенту ECI** не требуется событие для вывода из режима ожидания

Подробная семантика

- В случае если **хост ECI** не блокируется, он выходит из режима ожидания и незамедлительно запускает **клиента ECI**. Если хост блокируется, событие выхода из режима ожидания происходит позже, при первой же возможности. Требования к точности времени предлагаются в [b-ITU-T J Suppl. 7].

9.4.7.3.3 Сообщение reqPwrWakeupEvent

H→C reqPwrWakeupEvent() →
C→H resWakeupEvent()

- Это сообщение уведомляет **клиента ECI** об истечении времени таймера. **Клиент ECI** подтверждает этот **запрос** посредством **отклика** по завершении важного этапа обработки события выхода из режима ожидания.

Подробная семантика

- **Хост ECI** предпринимает попытки повторной отправки сообщения при успешном инициировании **клиента ECI** до тех пор, пока **клиент ECI** не отправит подтверждающее сообщение **resPwrWakeupEvent()**. Оповещение о событии отправляется в состоянии электропитания **PwrOn**, но откладывается в состоянии **PwrToStdby**.

Предварительные условия (запрос)

1) Таймер выхода из режима ожидания для **клиента ECI** предварительно установлен, и время истекло.
2) Событие еще не подтверждено **откликом**.
3) **Хост ECI** находится в состоянии электропитания **PwrOn**.

Постусловия (отклик)

1) **Хост ECI** прекращает отправку сообщений **reqPwrWakeupEvent**() на основе событий изменения режима энергопотребления по соответствующему **запросу**; см. **предварительное условие** 2).

9.4.7.4 Коды ошибок для API переключений режимов энергопотребления

Значения ошибок, связанных с конкретным интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются ниже в таблице 9.4.7.4-1.

Таблица 9.4.7.4-1 – Коды ошибок для API переключений режимов энергопотребления

Название	Значение	Описание
ErrPwrInfoNot	–256	**Клиент ECI** сообщает, что он не отправлял запрос на информирование о событиях изменений состояния электропитания

9.4.8 API для доступа к ресурсу настроек страны и языка

9.4.8.1 Введение

Интерфейс API для настроек страны и языка позволяет **клиенту ECI** или **хосту ECI** запрашивать актуальную информацию о настройках страны и языка **пользователя** у **хоста ECI** или **клиента ECI** соответственно. Сообщения для API настроек страны и языка перечисляются в таблице 9.4.8.1-1.

Таблица 9.4.8.1-1 – Сообщения API для настроек страны и языка

Сообщение	Тип	Направление	Маркер	Описание
reqHCountry	A	C→H	0x0	Запрашивает актуальную информацию о настройках предпочитаемой страны **хоста ECI**
reqCCountry	A	H→C	0x1	Запрашивает актуальную о настройках предпочитаемой страны **клиента ECI**
reqHLanguage	A	C→H	0x2	Запрашивает актуальную информацию о настройках предпочитаемого языка **хоста ECI**
reqCLanguage	A	H→C	0x3	Запрашивает актуальную информацию о настройках предпочитаемого языка **клиента ECI**

9.4.8.2 Определения сообщения API страны/языка

9.4.8.2.1 Сообщение настройки reqHCountry

C→H reqHCountry() →
H→C resHCountry setting (uint **iso_3166_country_code)**

- Это сообщение позволяет **клиенту ECI** запрашивать актуальную информацию о настройках страны, в которой в данный момент проживает **пользователь**, и получать **отклик** с сохраненной информацией о настройках страны от **хоста ECI**.

Определения параметра отклика

iso_3166_country_code: uint	Это поле содержит текущую информацию о настройках страны, имеющуюся у **хоста ECI**. Код страны представляет собой 24-битовое поле, в котором страна хоста обозначается тремя заглавными буквами согласно стандарту ISO 3166-1 alpha 3 [ISO 3166-1]. Каждый символ кодируется 8 битами согласно стандарту [ISO/IEC 8859-1]

Коды ошибок перечисляются в таблице 9.4.8.2.1-1.

Таблица 9.4.8.2.1-1 – Коды ошибок reqHCountry

Название	Описание
ErrCountryNotExists	**См. таблицу** 9.4.8.2.5-1

9.4.8.2.2 Сообщение настройки reqCCountry

H→C reqCCountry() →
C→H resCCountry setting (uint **iso_3166_country_code)**

- Это сообщение позволяет **хосту ECI** запрашивать актуальную информацию о настройках страны, в которой в данный момент проживает **пользователь**, и получать **отклик** с сохраненной информацией о настройках страны от **клиента ECI**.

Определения параметра отклика

iso_3166_country_code: uint	Это поле содержит текущую информацию о настройках страны, имеющуюся у **хоста ECI**. Код страны представляет собой 24-битовое поле, в котором страна хоста обозначается тремя заглавными буквами согласно стандарту ISO 3166-1 alpha 3 [ISO 3166-1]. Каждый символ кодируется 8 битами согласно стандарту [ISO/IEC 8859-1]

Коды ошибок перечисляются в таблице 9.4.8.2.2-1.

Таблица 9.4.8.2.2-1 – Коды ошибок reqCCountry

Название	Описание
ErrCountryNotExists	**См. таблицу** 9.4.8.2.5-1

9.4.8.2.3 Сообщение настройки reqHLanguage

H→C reqHLanguage(uint **iso_3166_language_code) →**
C→H resHLanguage setting()

- Это сообщение позволяет **клиенту ECI** запрашивать актуальную информацию о настройках языка, который в данный момент предпочитает **пользователь**, и получать **отклик** с сохраненной информацией о настройках языка от **хоста ECI**.

Определения параметра отклика

iso_3166_language_code: uint	Это поле содержит текущую информацию о настройках предпочитаемого языка, имеющуюся у **хоста ECI**. Код языка представляет собой 24-битовое поле, в котором язык обозначается тремя строчными буквами согласно стандарту [ISO 639-2]. Могут применяться оба стандарта: ISO 639-2/B и ISO 639-2/T. Каждый символ кодируется 8 битами согласно стандарту [ISO/IEC 8859-1]

Коды ошибок перечисляются в таблице 9.4.8.2.3-1.

Таблица 9.4.8.2.3-1 – Коды ошибок reqHLanguage

Название	Описание
ErrLanguageNotExists	См. таблицу 9.4.8.2.5-1

9.4.8.2.4 Сообщение настройки reqCLanguage

H→C reqCLanguage(uint **iso_3166_language_code) →**
C→H resCLanguage setting()

- Это сообщение позволяет **хосту ECI** запрашивать актуальную информацию о настройках языка, который в данный момент предпочитает **пользователь**, и получать **отклик** с сохраненной информацией о настройках языка от **клиента ECI**.

Определения параметра отклика

iso_3166_language_code: uint	Это поле содержит текущую информацию о настройках предпочитаемого языка, имеющуюся у **хоста ECI**. Код языка представляет собой 24-битовое поле, в котором язык обозначается тремя строчными буквами согласно стандарту [ISO 639-2]. Могут применяться оба стандарта ISO 639-2/B и ISO 639-2/T. Каждый символ кодируется 8 битами согласно стандарту [ISO/IEC 8859-1]

Коды ошибок перечисляются в таблице 9.4.8.2.4-1.

Таблица 9.4.8.2.4-1 – Коды ошибок reqCLanguage

Название	Описание
ErrLangageNotExists	**См. таблицу** 9.4.8.2.5-1

9.4.8.2.5 Коды ошибок для API настроек страны/языка

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются ниже в таблице 9.4.8.2.5-1.

Таблица 9.4.8.2.5-1 – Коды ошибок для API настроек страны/языка

Название	Значение	Описание
ErrCountryNotExists	–256	**Хост ECI** сообщает, что **пользователь** еще не указал страну, в которой проживает в данный момент
ErrLangageNotExists	–257	**Хост ECI** сообщает, что **пользователь** еще не указал предпочитаемый язык для любой связи через пользовательский интерфейс

9.5 Интерфейсы API для конкретных ресурсов хоста ECI

9.5.1 Список интерфейсов API для конкретных ресурсов хоста ECI

Рисунок 9.5.1-1 – Блок-схема интерфейсов API, определяемых в пункте 9.5

В таблице 9.5.1-1 перечисляются интерфейсы API, описываемые в пункте 9.8, а в таблице 9.5.1-1 показано расположение интерфейсов API, определяемых в пункте 9.5 с **архитектурой ECI**.

Таблица 9.5.1-1 – Список интерфейсов API, определяемых в пункте 9.5

Пункт	Наименование API	Описание
9.5.2	API усовершенствованной системы безопасности	Предоставляет **клиенту ECI** доступ к функциям усовершенствованной системы безопасности **хоста ECI**
9.5.3	API **смарт-карт**	Предоставляет **клиенту ECI** доступ к (дополнительному) устройству чтения **смарт-карт**
9.5.4	API обнаружения карусели передачи данных	Позволяет **клиенту ECI** обнаруживать карусель передачи данных в соответствии со стандартом DVB

9.5.2 API усовершенствованной системы безопасности

9.5.2.1 Введение

При загрузке **клиента ECI хост ECI** выделяет соответствующий сегмент усовершенствованной системы безопасности (типа **клиент ECI** или **микросервер**). Этот сегмент будет доступен в течение жизненного цикла **клиента ECI**. **Хост ECI** инициирует данный сегмент путем загрузки **цепочки сертификатов системы управления платформой**, содержащей открытый ключ **системы управления платформой**. Тем самым все дальнейшие значимые сеансы обмена с **AS-сегментом** привязываются к держателю секретного ключа **системы управления платформой**.

API усовершенствованной системы безопасности позволяет **клиенту ECI** взаимодействовать с функцией усовершенствованной системы безопасности оборудования **CPE**. Существует несколько типов обмена между **клиентом ECI** и AS-функцией. Как правило, обмен инициируется **клиентом ECI**. **Клиент ECI** получает сигнал по завершении длительных операций системы AS.

AS-сегмент поддерживает несколько сеансов, разрешая повторное использование сохраненной информации (состояние и конфигурацию) в **AS-сегменте** для дешифрования нескольких медиафайлов и **сеансов повторного шифрования**. **AS-сегмент** хранит один промежуточный ключ, называемый ключом связи высокого уровня секретности (LK_1) для каждого сеанса. Новые контрольные слова для сеансов можно быстро рассчитать на основе ключей LK_1.

AS-сегмент способен также рассчитывать секретный ключ аутентификации, который может использоваться для приложений **клиента ECI**, обеспечивая высокий уровень защиты при передаче секретной информации **клиенту ECI**.

Конфигурация **AS-сегмента** инициируется **клиентом ECI** и определяет режим его работы. **AS-сегмент** позволяет клиенту аутентифицировать его конфигурацию существует два основных режима аутентификации:

1) **режим многоступенчатой системы шифрования.** Аутентификация как часть расчета контрольного слова: конфигурация сегмента использовалась в расчете генерации контрольного слова для шифрования контента; та же самая информация необходима при расчете правильного контрольного слова для дешифрования контента, то есть косвенной аутентификации конфигурации.

2) **режим ключа аутентификации**. Аутентификация выполняется с помощью явной функции подтверждения при использовании данных проверки, которые могут быть сгенерированы только поставщиком **клиента ECI**. В сущности эта функция необходима для **AS-сегментов,** конфигурация которых предполагает повторное шифрование, поскольку за основу не может быть взято корректное дешифрование в качестве средства проверки.

В дополнение к вышеуказанным режимам **клиент ECI** может потребовать проведения повторной проверки при каждом инициировании сегмента посредством онлайн-аутентификации. В качестве альтернативы может выполняться офлайн-аутентификация. Для успешной аутентификации выбранный режим должен соответствовать данным, используемым для генерирования аутентификации, предоставляемой поставщиком.

API AS-системы разделяется на отдельные интерфейсы API, что позволяет отразить функциональные возможности использующих его **хостов ECI** и **клиента ECI**.

1) *Общий API AS-системы* – этот API определяет общие функциональные возможности AS-системы. Данный интерфейс должен поддерживаться всеми **хостами ECI** и **клиентами ECI**.

2) *API дешифрования AS-системы* – этот API определяет функциональные возможности AS-системы, связанные с дешифрованием. Данный интерфейс должен поддерживаться всеми **хостами ECI** и **клиентами ECI**, обладающими возможностями дешифрования.

3) *API экспорта AS-системы* – этот API определяет функциональные возможности AS-системы, связанные с экспортом. Данный интерфейс должен поддерживаться всеми **хостами ECI** и **клиентами ECI**, обладающими возможностями дешифрования и экспорта. **Хосты ECI**, поддерживающие экспорт, должны также поддерживать шифрование.

4) *API шифрования AS-системы* – этот API определяет функциональные возможности AS-системы, связанные с шифрованием. Этот интерфейс должен поддерживаться всеми хостами ECI и клиентами ECI, обладающими возможностями шифрования.

Действуют следующие ограничения:

- клиент ECI должен поддерживать либо дешифрование, либо шифрование, однако одновременная поддержка обеих функций не требуется.

Хост ECI и **клиент ECI** используют ресурс обнаружения интерфейса **хоста ECI** для того, чтобы предоставлять друг другу информацию относительно своих функциональных возможностей. **Хост ECI** назначает соответствующий сегмент согласно результатам обнаружения – сегмент шифрования для **клиентов ECI**, требующих шифрования, и AS-сегмент дешифрования для **клиентов ECI**, требующих дешифрования.

ПРИМЕЧАНИЕ. – Функции, обеспечивающие дополнительные функциональные возможности, могут существовать в различных интерфейсах API – общем API AS-системы и других конкретных API AS-системы.

В общем интерфейсе API AS-системы сообщения должны поддерживаться только **хостом ECI**, поскольку это необходимо для отражения функциональных возможностей **хоста ECI** (поддержки дешифрования, экспорта и шифрования).

Сообщения интерфейсов API AS-системы определяются с точки зрения функций AS-системы, описываемых в пунктах 8.2.4 и 9.9 [ITU-T J.1014]. В пункте 8.2.4.1 [ITU-T J.1014] приводится обзор

функций AS-системы. В определениях, указываемых в [ITU-T J.1014], опущен первый параметр – slotId; он предоставляется **хостом ECI**.

Многие определения типов и значений параметров, используемых в этом определении API, приводятся в [ITU-T J.1014]. Коды ошибок для этого API определяются в [ITU-T J.1014]; в настоящей Рекомендации в конкретном плане не приводится их перечень для каждого сообщения. Коды ошибок для значений параметров соответствуют нумерации последовательности параметров, определяемой упомянутыми в [ITU-T J.1014] функциями, в которых, как правило, присутствует один дополнительный параметр (slotId).

9.5.2.2 Определения сообщений общего API усовершенствованной системы безопасности

9.5.2.2.1 Общие сведения

Общий API усовершенствованной системы безопасности предоставляет сообщения, перечисляемые в таблице 9.5.2.2.1-1.

Таблица 9.5.2.2.1-1 – Общие сообщения усовершенствованной системы безопасности

Сообщение	Тип	Направление	Маркер	Описание
reqAsInitSlot	A	C→H	0x0	Инициирует **AS-сегмент**
callAsNextKeySession	S	C→H	0x1	Переход к следующему случайному ключу для сеанса
reqAsStopSession	A	C→H	0x2	Остановка сеанса
reqAsLoadSlotLk	A	C→H	0x3	Расчет ключа связи высокого уровня (LK1)
reqAsComputeAkClient	A	C→H	0x4	Расчет ключа аутентификации для приложений **клиента ECI**
reqAsClientChalResp	A	C→H	0x5	Применение ключа аутентификации **клиента ECI** к данным и возврат результата
getAsSlotRk	S	C→H	0x6	Получение случайного значения ключа для **AS-сегмента**
getAsSessionRk	S	C→H	0x7	Получение случайного значения ключа для сеанса
getAsSessionLimitCounter	S	C→H	0x8	Получение текущего предельного значения счетчика для сеанса
setAsSessionLimitEvent	S	C→H	0x9	Установка предельного значения для отправки сообщения reqAsEventSessionLimit **клиенту ECI**
reqAsEventSessionLimit	A	H→C	0xA	Отправка события **клиенту ECI** по достижении предельного значения для оставшихся единиц
getAsClientRnd	S	C→H	0xB	Получение нового случайного числа для приложений **клиента ECI**
getAsSC	S	C→H	0xC	Получение текущего статуса поля управления скремблированием контента в сеансе
reqAsEventSC	A	H→C	0xD	Событие сообщения об изменении поля управления скремблированием в сеансе
getChipsetId	S	C→H	0xE	Получение значения параметра ChipsetID **блока лестницы ключей**
getImageTargetId	S	C→H	0xF	Получение значения параметра ECI_Image_Target_Id оборудования CPE

9.5.2.2.2 Сообщение reqAsInitSlot

C→H reqAsInitSlot(uint **slotVersion,** uint **slotMode** →
H→C resAsInitSlot()

- Это сообщение инициирует сегмент с различными общими параметрами.

Определения параметра запроса

slotVersion: uint	Версия набора функций сегмента, определяемая в [ITU-T J.1014].
slotMode: uint	Основной режим работы для сегмента; см. [ITU-T J.1014].

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsInitSlot, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметров slotId и POPKchain.

9.5.2.2.3 Сообщение callAsNextKeySession

C→H callAsNextKeySession(uint **sessionId**)

- Это сообщение вызывает переход к следующему случайному ключу сеанса.

Определения параметра запроса

sessionId: uint	Сеанс, для которого объявляется переход к следующему случайному ключу

Семантическое описание

- Это сообщение эквивалентно сообщению AS-системы callAsNextKeySession, определяемому в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.4 СообщениереqAsStopSession

C→H reqAsStopSession(uint **sessionId**) →
H→C resAsStopSession()

- Это сообщение останавливает сеанс **AS-сегмента**.

Определения параметра запроса

sessionId: uint	Идентификатор останавливаемого сегмента

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsStopSession, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.5 Сообщение reqAsLoadSlotLk

C→H reqAsLoadSlotLk(uint **sessId**, InputV **inputV**, ulong **spkUri,** uchar **spkIndx**) → **H→C resAsLoadSlotLk**()

- Это сообщение рассчитывает ключ связи высокого уровня LK_1, который затем используется для расчета контрольных слов.

Определения параметра запроса

sessId: uint	Идентификатор инициируемого сегмента
inputV: InputV	Сообщение содержит ключ связи LK_1, зашифрованный открытым ключом чипсета и защищенный подписью секретного ключа отправителя
spkUri: ulong	Правила использования вектора SPK для последующего расчета контрольного слова приведены в [ITU-T J.1014]
spkIndx: uchar	Индекс, определяющий расположение SPK AS-сегмента в векторе SPK, который используется для последующего расчета контрольного слова, см. в разделе 7 [ITU-T J.1014]

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsLoadSlotLk, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.
- **Хост ECI** также предоставляет функцию reqAsDecoupleDecryptSession [ITU-T J.1014], если останавливается сеанс дешифрования сегмента системы AS, который ранее был сопряжен с другим сеансом дешифрования сегмента системы AS (см. пункт 9.5.2.3.1).

9.5.2.2.6 Сообщение 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 ()

- Это сообщение рассчитывает ключ аутентификации для использования **клиентом ECI**.

Определения параметра запроса

inputV: InputV	Сообщение содержит зашифрованное открытым ключом чипсета и защищенное подписью секретного ключа отправителя значение *r*, используемое для расчета ключа аутентификации
nSpk: uint	Количество значений в векторе SPK, см. [ITU-T J.1014]
spkIndx: uchar	Индекс, определяющий расположение SPK AS-сегмента в векторе SPK, значение POPK AS-сегмента в векторе POPK и значение slotConfig AS-сегмента в векторе clCnf, используемом для расчета ключа аутентификации клиента, см. [ITU-T J.1014]
spk[16]: PubKey	Вектор **открытого ключа отправителя**, используемый для расчета ключа аутентификации клиента; см. [ITU-T J.1014]
popk[16]: PubKey	Вектор открытого ключа системы управления платформой, используемый для расчета ключа аутентификации клиента; см. [ITU-T J.1014]
akCnf[16]: SessionConfig	Вектор конфигурации сеанса клиента, используемый для расчета ключа аутентификации клиента; см. [ITU-T J.1014]
spkUri: ulong	Правила использования вектора SPK для последующего расчета контрольного слова см. в [ITU-T J.1014]
XT[32]: uchar	Значение поля расширения, используемое для расчета ключа аутентификации клиента; см. [ITU-T J.1014]. Значение по умолчанию равно { 0x00 }
online: bool	Если значение true, случайный ключ сегмента используется для расчета ключа аутентификации, принудительно запуская новый расчет этого ключа поставщиком

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsComputeAkClient, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.7 Сообщение reqAsClientChalResp

C→H reqAsClientChalResp(uchar **challenge**[16]);→
H→C reqAsClientChalResp(uchar **response**[16])

- Это сообщение использует ключ аутентификации клиента, рассчитанный с помощью сообщения reqAsComputeAkClient (определяемого в [ITU-T J.1014]), для дешифрования 128-битового входного параметра запроса ключа и формирования 128-битового выходного параметра отклика.

Определения параметра запроса

challenge[16]: uchar	128-битовые входные данные, подлежащие дешифрованию ключом аутентификации клиента

Определения параметра отклика

response[16]: uchar	128-битовые дешифрованные выходные данные

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsClientChalResp, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId, а сообщение-**отклик** передает выходной параметр "отклик".

9.5.2.2.8 Сообщение getAsSlotRk

C→H SymKey **getAsSlotRk**()

- Это сообщение считывает случайный ключ для сеанса **AS-сегмента клиента ECI**.

Семантическое описание

- Это сообщение эквивалентно функции **AS**-системы getAsSlotRk, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.9 Сообщение getAsSessionRk

C→H SymKey **getAsSessionRk**(uint **sessionId,** uint **rkIndx)**

- Это сообщение считывает текущий (rkIndx==0) или следующий (rkIndx==1) случайный ключ для сеанса **клиента ECI** с идентификатором sessionId.

Определения параметра запроса

sessionId: uint	Id сеанса, для которого должен быть получен случайный ключ для сеанса
rkIndx: uint	Определяет, должен ли быть получен текущий (rkIndx==0) или следующий (rkIndx==1) случайный ключ для сеанса

Семантическое описание

- Это сообщение эквивалентно сообщению AS-системы getAsSessionRk, определяемому в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.10 Сообщение getAsSessionLimitCounter

C→H ulong **getAsSessionLimitCounter**(uint **sessionId**)

- Это сообщение возвращает предельное значение счетчика идентификатора sessionId **клиента ECI**.

Семантическое описание

- Эта функция эквивалентна функции AS-системы getAsSessionLimitCounter, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

Определения параметра запроса

sessionId: uint	Id сеанса, для которого должно быть получено предельное значение счетчика для сеанса

9.5.2.2.11 Сообщение setAsSessionLimitEvent

C→H ulong **setAsSessionLimitEvent** (uint **sessionId,** ulong **eventLimit**)

- Это сообщение устанавливает предельное значение счетчика eventLimit для limitCounter сеанса **клиента ECI** с идентификатором sessionId, чтобы сообщение reqAsEventSessionLimit было возвращено **клиенту ECI**.

Определения параметра запроса

sessionId: uint	Id сеанса, для которого должно быть установлено предельное значение eventLimit сеанса
eventLimit: ulong	Предельное значение события, подлежащее установке

Семантическое описание

- Эта функция эквивалентна функции AS-системы setAsSessionLimitEvent, определенной в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.12 Сообщение reqAsEventSessionLimit

H→C reqAsEventSessionLimit (uint **sessionId**)
C→H res**AsEventSessionLimit ()**

- Это сообщение возвращает предельное значение счетчика идентификатора sessionId **клиента ECI**.

Определения параметра отклика

sessionId: uint	Идентификатор сеанса, генерирующего событие eventLimit

Семантическое описание

- Эта функция эквивалентна функции AS-системы reqAsEventSessionLimit, определяемой в [ITU-T J.1014]; при этом **хост ECI** удаляет параметр slotId.

9.5.2.2.13 Сообщение getAsClientRnd

C→H SymKey **getAsClientRnd**()

- Это сообщение возвращает 128-битовое случайное число.

Семантическое описание

- Эта функция эквивалентна сообщению AS-системы getAsClientRnd, определяемой в [ITU-T J.1014].

9.5.2.2.14 Сообщение getAsSC

C→H uint **getAsSC**(uint **sessionId**)

- Это сообщение возвращает текущий статус поля управления скремблированием контента в сеансе.

Определения параметра запроса

sessionId: uint	Id сеанса, для которого должно быть получено текущее поле управления скремблированием

Семантическое описание

- Эта функция эквивалентна функции AS-системы getAsSC, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.2.15 Сообщение reqAsEventSC

H→C reqAsEventSC(uint **sessionId**; uint **scramblingControlField**)
C→H res**AsEventSC()**

- Это сообщение указывает на изменение поля управления скремблированием в сеансе с идентификатором sessionId.

Определения параметра отклика

sessionId: uint	Id сеанса, во время которого произошло изменения поля статуса скремблирования
scramblingControlFieId: uint	Новое значение поля статуса скремблирования. Определение и семантика значений приведены в пункте 9.9 [ITU-T J.1014]

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsEventSC, определяемой в [ITU-T J.1014]; при этом **хост ECI** удаляет значение параметра slotId.

9.5.2.2.16 Сообщение getChipsetId

C→H ulong **getChipsetId**()

- Это сообщение возвращает значение параметра ChipsetID **блока лестницы ключей**, как определено в [ITU-T J.1014].

9.5.2.2.17 Сообщение getImageTargetId

C→H ECI_Image_Target_Id **getImageTargetId**()

- Это сообщение возвращает значение параметра ECI_Image_Target_Id оборудования CPE, как определено в таблице 6.2.2.2-1.

9.5.2.3 Определения сообщений API дешифрования усовершенствованной системы безопасности

9.5.2.3.1 Общие сведения

Идентификатор API дешифрования усовершенствованной системы безопасности предоставляет сообщения, перечисляемые в таблице 9.5.2.3.1-1.

Два сеанса дешифрования могут быть совмещены, что позволит использовать разные контрольные слова для дешифрования двух потоков контента, которые должны после дешифрования обрабатываться как единый контентный элемент.

ПРИМЕР. Вещание на спортивном канале может проводиться с подключением нескольких звуковых каналов; при этом звуковой канал для конкретного языка доступен только при наличии специальной подписки, позволяющей дешифровать данный канал. Только один сеанс может быть соединен с другим сеансом.

Таблица 9.5.2.3.1-1 – Сообщения дешифрования усовершенствованной системы безопасности

Сообщение	Тип	Направление	Маркер	Описание
reqAsAStartDecryptSession	A	H→C	0x0	Запуск сеанса дешифрования в **AS-сегменте клиента ECI**
reqAsComputeDecrCw	A	H→C	0x1	Расчет контрольного слова дешифрования
reqAsAuthDecrSlotConfig	A	H→C	0x2	Аутентификация конфигурации сегмента с применением соответствующих механизмов (режим дешифрования)

9.5.2.3.2 Сообщение reqAsStartDecryptSession

C→H reqAsAStartDecryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config,** ScrambleMode **sm**) → **H→C resAsAStartDecryptSession(**uint **sessionId)**

- Это сообщение запускает сеанс дешифрования в **AS-сегменте клиента ECI**.

Определения параметра запроса

mh: ushort	**Указатель медиаданных**, для которого дешифруется контент (используемый **хостом ECI** для привязки контента, подлежащего дешифрованию, к ресурсу дешифрования, выделенному для данного сеанса)
spk: PubKey	Открытый ключ отправителя для данного сеанса
config: SessionConfig	Конфигурация сеанса
sm: ScrambleMode	Применяемый режим дескремблирования. Определение приведено в таблице 9.5.2.3.2-1. См. примечание
ПРИМЕЧАНИЕ. – Информация параметра sm не должна противоречить параметру cwUri последующего сообщения reqAsComputeDecrCw.	

Таблица 9.5.2.3.2-1 – Определение ScrambleMode

```
typedef ScrambleMode {
  uchar    modeRef;
  uchar    mode[16] ;
} ScrambleMode;
```

Определение **modeRef** приводится в таблице 9.5.2.3.2-2.

Таблица 9.5.2.3.2-2 – Определение modeRef

Название	Значение	Описание
ScrambleModeHost	0x01	Хост выбирает режим (де)скремблирования на основе стандартной или проприетарной информации
ScrambleModeDvb	0x02	Используется определение DVB для режима скремблирования. Значение байта 0 поля режима идентично значению, определяемому в поле scrambling_mode дескриптора Scrambling_descriptor, как определяется в [IEC 62766-5-2]. Байт 1 имеет следующее значение для байта 0: 0x02, 0x03 и 0x10 (то есть режимы DVB CSA1/2, DVB CSA3 для дескремблирования и режим DVB-CISSA, версия 1). Значение==0x01 – де(скремблирование) в режиме TS. Значение==0x02 – де(скремблирование) в режиме PES. Все другие значения резервируются; все неиспользуемые байты поля режима резервируются. См. примечание 1
ScrambleModeCencEnum	0x03	Режим скремблирования определяется в [ITU-T T.871], или байт 0 поля режима определяется как: значение==0x01 – режим CENC CTR; значение==0x02 – режим CENC CBC. Другие значения для байта 0 резервируются. При значениях байта 0, определенных выше, байт 1 указывает на подсхему: значение==0x01 – определяется хостом, для шифрования выбирается одно из значений, определенных ниже; значение==0x02 – полное шифрование сегмента, как определяется в [W3C GIF V89a]; значение==0x03 – шифрование подвыборки, как определено в [W3C PNG]. Другие значения для байта 1 резервируются. При других значениях байта 0 байт 1 резервируется. Байты 2–15 резервируются. См. примечание 2
RFU	Прочее	Зарезервировано для использования в будущем
ПРИМЕЧАНИЕ 1. – Хост ECI должен как минимум поддерживать режим DVB CSA1/2, DVB CSA3 для дескремблирования и режим DVB-CISSA (версия 1) для скремблирования и дескремблирования. ПРИМЕЧАНИЕ 2. – Клиент ECI или (если разрешено) хост ECI могут выбирать режим скремблирования для шифрования, подходящий для приложения; особо учитываются приложения потокового типа, которые, как правило, используют полное шифрование сегмента CBC, и приложения, сохраняющие данные, которые, как правило, используют режим CTR. Полезным может оказаться шифрование подвыборок.		

Определения параметра отклика

sessionId: uint	Идентификатор созданного сеанса

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsAStartDecryptSession, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId, а результат sessionId возвращается в сообщении-**отклике**.

Хост ECI предоставляет также функцию reqAsCoupleDecryptSession [ITU-T J.1014] при запуске второго сеанса дешифрования сегмента системы AS для того же **указателя медиаданных** таким образом, что сеансы дешифрования соединяются – второй сеанс присоединяется к первому.

9.5.2.3.3 Сообщение reqAsComputeDecrCw

C→H reqAsComputeDecrCw(int **sessionId**, ulong **cwUri,** uint **nSpk**, uint **nElk**, SymKey **elk**[24], PubKey **spk**[16], PubKey **popk**[16], SessionConfig **config**[16], uchar **XT**[32], uint **rkIndx,** Field2 **field2,** uint **cwIndx**) →
H→C resAsComputeDecrCw ()

- Это сообщение рассчитывает контрольное слово дешифрования.

Определения параметра запроса

sessionId: int	Id сеанса, для которого рассчитывается контрольное слово
cwUri: ulong	Значение cwUri определяет приложения контрольного слова. Значения cwUri определяются в пункте 7.5 [ITU-T J.1014]
nSpk: uint	Количество значений SPK в векторе SPK
nElk: uint	Количество значений Elk в векторе ELK
elk[24]: SymKey	Вектор симметрично зашифрованных значений ключа, которые должны быть дешифрованы при помощи многоступенчатой системы ключей. Значение elk[nElk-2] – это данные, введенные в поле field1 для аутентификации свойства контента, как определяется в пункте 8.2.3 [ITU-T J.1014], с использованием функции, определенной в пункте 8.2.4.7 [ITU-T J.1014]
spk[16]: PubKey	Вектор открытых ключей отправителя, определяемый в пункте 7.5 [ITU-T J.1014]
popk[16]: PubKey	Вектор открытых ключей оператора платформы, определяемый в пункте 7.5 [ITU-T J.1014]
config[16]: SessionConfig	Вектор конфигураций сеансов клиента, определяемый в пункте 7.5 [ITU-T J.1014]
XT[32]: uchar	Дополнительные входные данные для механизма контрольного слова, определяемые в пункте 7.5 [ITU-T J.1014]
rkIndx: uint	Определяет, используется ли текущий (rkIndx==0) или следующий (rkIndx==1) случайный ключ сеанса при расчете контрольного слова
field2: Field2	Контент, содержащий большой объем данных и не аутентифицированный в поле field1, как определяется в пункте 8.2.3 [ITU-T J.1014]
cwIndx: uint	Индекс рассчитываемых контрольных слов: 0 для четного и 1 для нечетного контрольного слова; значение для дешифрования на основе файлов отсутствует

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsComputeDecrCw, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.3.4 Сообщение 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 ()

- Это сообщение выполняет аутентификацию конфигурации сегмента с применением соответствующих механизмов (режим дешифрования).

Определения параметра запроса

sessionId: uint	Id сеанса, для которого выполняется аутентификация сегмента
inputV: InputV	Сообщение содержит зашифрованное открытым ключом чипсета и защищенное подписью секретного ключа отправителя значение r, используемое для расчета ключа аутентификации в целях аутентификации конфигурации **AS-сегмента**
nSpk: uchar	Количество значений SPK в векторе SPK
spkIndx: uint	Индекс, определяющий расположение SPK AS-сегмента в векторе SPK, значение POPK AS-сегмента в векторе POPK и значение slotConfig AS-сегмента в векторе clCnf, используемый для расчета ключа аутентификации клиента, см. [ITU-T J.1014]
spk[16]: PubKey	Вектор открытых ключей отправителя, определяемый в пункте 7.5 [ITU-T J.1014
popk[16]: PubKey	Вектор открытых ключей оператора платформы, определяемый в пункте 7.5 [ITU-T J.1014]
cnf[16]: SessionConfig	Вектор конфигураций клиента, определяемый в пункте 7.5 [ITU-T J.1014]
spkUri: ulong	Правила использования вектора SPK для последующего расчета ключа аутентификации см. в [ITU-T J.1014]
XT[32]: uchar	Значение поля расширения, используемое для расчета ключа аутентификации клиента; см. [ITU-T J.1014]. Значение по умолчанию равно { 0x00 }
online: bool	Если значение true, случайный ключ сегмента используется для расчета ключа аутентификации, принудительно запуская новый расчет этого ключа поставщиком
verifier[16]: uchar	Значение, при котором reqAsAuthDecrSlotConfig выполняет аутентификацию конфигурации сегмента

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsAuthDecrSlotConfig, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.4 API экспорта усовершенствованной системы безопасности

9.5.2.4.1 Общие сведения

Общий API экспорта усовершенствованной системы безопасности предоставляет сообщения, определяемые в таблице 9.5.2.4.1-1.

Таблица 9.5.2.4.1-1 – Сообщения экспорта усовершенствованной системы безопасности

Сообщение	Тип	Направление	Маркер	Описание
reqAsExportConnSetup	A	C→H	0x0	Настройка **соединения экспорта** от сеанса дешифрования до сеанса шифрования
reqAsExportConnEnd	A	C→H	0x1	Завершает текущий сеанс экспорта

9.5.2.4.2 Сообщение reqAsExportConnSetup

C→H reqAsExportConnSetup(uint **sessId,** ushort **expMh,** uint **grpIndx**; CertSerialChain **expCh,** CertSerialChain **impCh,** CertSerialChain **auth**[]) →
H→C resAsExportConnSetup()

- Это сообщение настраивает соединение усовершенствованной системы безопасности от сеанса дешифрования до сеанса **указателя медиаданных** экспорта.

Определения параметра запроса

sessId: uint	Идентификатор сеанса экспорта **AS-сегмента клиента ECI**
expMh: ushort	Идентификатор **указателя медиаданных** экспорта, который используется для шифрования дешифрованного контента в AS-сеансах
grpIndx: uint	Индекс сохранения соединения сеанса экспорта; допустимые значения равны 0 или 1. Этот параметр может использоваться для изменения аутентификации **соединения экспорта** на **микросервере** (например, для прогнозирования предстоящей перенастройки идентификатора группы экспорта в потоке)
expCh: CertSerialChain	**Цепочка экспорта** для **клиента ECI**
impCh: CertSerialChain	**Цепочка импорта** для шифрования/импорта **клиента ECI**
auth[]: CertSerialChain	**Сертификаты** авторизации для **цепочки импорта**

Семантическое описание

- Это сообщение эквивалентно функции **AS**-системы reqAsExportConnSetup, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметров slotId, impSlotId и ImpSessId. **Хост ECI** использует **указатель медиаданных** сеанса экспорта для подключения AS-сеанса дешифрования к соответствующему AS-сеансу шифрования, то есть предоставляет параметры impSlotId и impSessId в AS-функцию reqAsExportConnSetup, определяемую в [ITU-T J.1014].

9.5.2.4.3 Сообщение reqAsExportConnEnd

C→H reqAsExportConnEnd(ushort **expMh)** →
H→C resAsExportConnEnd()

- Это сообщение завершает текущий сеанс экспорта.

Определения параметра запроса

expMh: ushort	Экспорт сеанса **указателя медиаданных** из ряда AS-сеансов, для которого обмен контентом должен быть прекращен

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsExportConnEnd, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметров slotId и sessionId, связанных с expMh.

9.5.2.5 API шифрования усовершенствованной системы безопасности

9.5.2.5.1 Общие сведения

API шифрования усовершенствованной системы безопасности предоставляет сообщения, перечисляемые в таблице 9.5.2.5.1-1.

Таблица 9.5.2.5.1-1 – Сообщения шифрования усовершенствованной системы безопасности

Сообщение	Тип	Направление	Маркер	Описание
reqAsStartEncryptSession	A	C→H	0x0	Запуск сеанса шифрования
reqAsComputeEncrCw	A	C→H	0x1	Расчет контрольного слова шифрования
reqAsAuthEncrSlotConfig	A	C→H	0x2	Аутентификация конфигурации сегмента и параметров шифрования с применением соответствующих механизмов (режим шифрования)
reqAsLdUssk	A	C→H	0x3	Загрузка секретного ключа **микросервера**
reqAsMInikLk1	A	C→H	0x4	Вычисление асимметричного сообщения инициирования **микроклиента**
reqAsEventCpChange	A	H→C	0x5	Событие сообщения об изменении свойств импортированного контента во время сеанса шифрования
setAsPermitCPChange	S	C→H	0x6	Включение и отключение изменений свойств импортированного контента путем выбора контрольного слова при шифровании во время сеанса шифрования
setAsSC	S	C→H	0x7	Установка поля управления скремблированием для зашифрованного контента во время сеанса шифрования

9.5.2.5.2 Определение целевой цепочки клиентов

Микросерверы могут использовать **систему обработки сертификатов** для построения устойчивой системы асимметричной аутентификации клиента. Интерфейс **ECI** определяет цепочки сертификатов, что позволяет выполнять аутентификацию **микроклиента**. Подобные целевые цепочки используются в качестве входных данных для сообщения reqAsMInikLk1.

Цепочки сертификатов должны соответствовать пункту 5.4.1. В процессе участвуют два типа **сертификатов**:

- сертификат микроклиента обеспечивает аутентификацию одиночного микроклиента; открытый ключ сертификата идентичен открытому ключу чипсета оборудования CPE микроклиента в случае, если микроклиентом является клиент ECI;
- сертификат целевой группы обеспечивает аутентификацию одной или нескольких целевых групп или сертификатов **микроклиентов**.

Операторы **микросистем DRM** могут использовать механизм **списка аннулирования ECI** для безопасного управления развертыванием аутентифицированных **микроклиентов** для сервера.

ПРИМЕЧАНИЕ. – Обслуживание **списков аннулирования** является частным делом оператора **микросистемы DRM**.

Идентификатор **сертификата** целевой группы определяется в таблице 9.5.2.5.2-1.

Таблица 9.5.2.5.2-1 – Определение идентификатора целевой группы

Синтаксис	Количество битов	Мнемоника
ECI_Target_Group_Id {		
padding(4)		
type	4	uimsbf
target_group_id	20	uimsbf
target_group_version	8	uimsbf
}		

Семантика

type: integer	Значение согласно таблице 5.1.3-1
target_group_id: integer	Номер **целевой группы**, уникальный в контексте **родительского объекта**
target_group_version: integer	Возрастает при изменении **сертификата** микрогруппы

Идентификатор **сертификата** микроклиента определяется в таблице 9.5.2.5.2-2.

Таблица 9.5.2.5.2-2 – Определение идентификатора микроклиента

Синтаксис	Количество битов	Мнемоника
ECI_Micro_Client_Id {		
padding(4)		
type	4	uimsbf
micro_client_id	20	uimsbf
micro_client_version	8	uimsbf
}		

Семантика

type: integer	Значение согласно таблице 5.1.3-1
micro_client_id: integer	Номер **целевой группы**, уникальный в контексте **родительского объекта**
micro_client_version: integer	Возрастает при изменении **сертификата** микрогруппы

9.5.2.5.3 Сообщение reqAsStartEncryptSession

C→H reqAsStartEncryptSession(ushort **mh**, PubKey **spk**, SessionConfig **config**, uint **nEncr**, PubKey **encrSpk[MaxSpkEncr]**, PubKey **encrPopk[MaxSpkEncr]**, ulong **encrCwUri**)→ **H→C resAsStartEncryptSession()**

- Это сообщение запускает сеанс шифрования.

Определения параметра запроса

mh: ushort	Идентификатор **указателя медиаданных** зашифрованного контента, для которого создается сеанс шифрования
spk: PubKey	Открытый ключ отправителя, используемый для аутентификации отправителя и сообщения, зашифрованного AS-системой посредством ключа связи LK
config: SessionConfig	Конфигурация сеанса
nEncr: uint	Количество дополнительных значений SPK (и POPK), определяемых для шифрования и возможного последующего дешифрования. Максимальное значение равно MaxEncr; см. [ITU-T J.1014]
encrSpk: PubKey[]	Вектор с дополнительными значениями SPK для шифрования
encrPopk: PubKey[]	Вектор с дополнительными значениями POPK для шифрования
encrCwUri: ulong	Значение CWURI, используемое для шифрования; см. пункт 8.2.2 [ITU-T J.1014]

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsStartEncryptSession, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId. **Хост ECI** извлекает параметры, importSlotId и importCeancId из значения mh.

ПРИМЕЧАНИЕ. – Сообщение-**отклик** возвращает идентификатор нового сеанса, создаваемый при отсутствии ошибок.

9.5.2.5.4 Сообщение 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()

- Это сообщение рассчитывает контрольное слово шифрования.

Определения параметра запроса

sessId: int	Идентификатор сеанса для расчета контрольного слова
cwUri: ulong	Значение cwUri определяет приложения контрольного слова. Значения cwUri определяются в пункте 7.5 [ITU-T J.1014]
nElk: uint	Количество значений Elk в векторе ELK
elk[24]: SymKey	Вектор симметрично зашифрованных значений ключа, которые должны быть дешифрованы при помощи многоступенчатой системы ключей. Значение elk[nElk-2] – это данные, введенные в поле field1 для аутентификации свойства контента, как определяется в пункте 8.2.3 [ITU-T J.1014], с использованием функции, определенной в пункте 8.4.2.6 [ITU-T J.1014]
XT[32]: uchar	Дополнительные входные данные для механизма контрольного слова, как определяется в пункте 7.5 [ITU-T J.1014]
rkIndx: uint	Определяет, используется ли текущий (rkIndx==0) или следующий (rkIndx==1) случайный ключ сеанса при расчете контрольного слова
field2: Field2	Контент, содержащий большой объем данных и не аутентифицированный в поле field1, как определяется в пункте 8.2.3 [ITU-T J.1014]
cwIndx: uint	Индекс рассчитываемых контрольных слов: 0 для четного и 1 для нечетного контрольного слова; значение для дешифрования на основе файлов отсутствует

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsComputeEncrCw, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.5 Сообщение reqAsAuthEncrSlotConfig

C→H reqAsAuthEncrSlotConfig(uint **sessId**, InputV **inputV**, uchar **XT**[32], bool **online,** uchar **verifier[16]**) →
H→C resAsAuthEncrSlotConfig()

- Это сообщение выполняет аутентификацию конфигурации сегмента с применением соответствующих механизмов (режим шифрования).

Определения параметра запроса

sessId: uint	Идентификатор сеанса, для которого выполняется аутентификация конфигурации.
inputV: InputV	Сообщение содержит зашифрованное открытым ключом чипсета и защищенное подписью секретного ключа отправителя значение r, используемое для расчета ключа аутентификации в целях аутентификации конфигурации **AS-сегмента**
XT[32]: uchar	Дополнительные входные данные для механизма контрольного слова, как определяется в пункте 7.5 [ITU-T J.1014]
online: bool	Если значение true, случайный ключ сегмента используется для расчета ключа аутентификации, принудительно запуская новый расчет этого ключа поставщиком
verifier[16]: uchar	Значение, при котором reqAsAuthDecrSlotConfig выполняет аутентификацию конфигурации сегмента

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsAuthEncrConfig, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.6 Сообщение reqAsLdUssk

C→H reqAsLdUssk(uint **sessId**, InputV **inputV,** uchar **XT**[32], bool **online,** uchar **mUssk**[NUSSK])→
H→C resAsLdUssk()

- Это сообщение загружает секретный ключ **микросервера** при асимметричной аутентификации **клиентов ECI**, способных декодировать контент.

Определения параметра запроса

sessId: uint	Идентификатор сеанса, для которого будет загружен секретный ключ **микросервера**
inputV: InputV	Сообщение содержит зашифрованное открытым ключом чипсета и защищенное подписью секретного ключа отправителя значение r, используемое для расчета ключа аутентификации в целях дешифрования загружаемого секретного ключа **микросервера**
XT[32]: uchar	Дополнительные входные данные для механизма контрольного слова, как определяется в пункте 7.5 [ITU-T J.1014]
online: bool	Если значение true, случайный ключ сегмента используется для расчета ключа аутентификации, принудительно запуская новый расчет этого ключа поставщиком
mUssk[NUSSK]: uchar	Секретный ключ зашифрованного **микросервера**

Семантическое описание

- Эта функция эквивалентна функции AS-системы reqAsLdUssk, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.7 Сообщение reqAsMInikLk1

C→H reqAsMInikLk1(uint **sessId**, ECI_Certificate_Chain **ClCPK**) →
H→C resAsMInikLk1(InputV **inputV)**

- Это сообщение рассчитывает асимметричное сообщение инициирования **микроклиента**.

Определения параметра запроса

sessId: uint	Идентификатор сеанса, для которого будет загружен секретный ключ **микросервера**
ClCPK: ECI_Certificate_Chain	Целевая **цепочка сертификатов**, определяемая в пункте 9.5.2.5.2, для загрузки открытого ключа чипсета микроклиента, который будет использоваться для шифрования секретного ключа сеанса между **микросервером** и **микроклиентом**

Определение параметра отклика

inputV: InputV	Ключ сеанса microDRM, зашифрованный открытым ключом чипсета микроклиента и подписанный секретным ключом **микросервера**. Может использоваться микроклиентом в качестве сообщения для загрузки общего ключа связи LK_1 сеанса

Семантическое описание

- Эта функция эквивалентна функции AS-системы reqAsMInikLk1, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.8 Сообщение reqAsEventCpChange

H→C reqAsEventCpChange(int **sessionId**)

- Это сообщение отправляет запрос об изменении свойств импортируемого контента во время сеанса шифрования.

Определение параметра запроса

sessionId: int	Сеанс шифрования, во время которого происходит изменение свойств импортируемого контента

Семантическое описание

- Это сообщение эквивалентно функции AS-системы reqAsEventCpChange, определяемой в [ITU-T J.1014]; при этом **хост ECI** удаляет значение параметра slotId.

9.5.2.5.9 Сообщение setAsPermitCPChange

C→H setAsPermitCPChange(int **sessionId**; bool **permit**)

- Это сообщение инициирует изменение свойств импортируемого контента во время сеанса шифрования.

Определения параметра запроса

sessionId: int	Сеанс шифрования для разрешения автоматической перенастройки контрольного слова при изменении свойств контента, которое произойдет или ожидается
permit: булева переменная	Значение true говорит о том, что разрешение выдано; False означает, что разрешение не выдано

Семантическое описание

- Эта функция эквивалентна функции AS-системы setAsPermitCPChange, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.10 Сообщение setAsSC

C→H setAsSC(int **sessionId**, uint **scramblingControlField**)

- Это сообщение устанавливает следующее значения поля управления скремблированием в сеансе шифрования.

Определения параметра запроса

sessionId: int	Сеанс шифрования, для которого поле управления скремблированием должно быть задано, то есть должно использоваться в первой возможной точке изменения в потоке
scramblingControlFieId: uint	Значение поля управления скремблированием; допустимые значения и их расшифровка приводятся в пункте 9.9 [ITU-T J.1014]

Семантическое описание

- Эта функция эквивалентна функции AS-системы setAsSC, определяемой в [ITU-T J.1014]; при этом **хост ECI** предоставляет значение параметра slotId.

9.5.2.5.11 Коды ошибок для API усовершенствованной системы безопасности (AS)

Коды ошибок для интерфейсов API системы AS определяются в пункте 8.2.4.15 [ITU-T J.1014].

9.5.3 API смарт-карты

9.5.3.1 Введение

Интерфейс **ECI** позволяет **клиентам ECI** взаимодействовать с единым съемным локальным защищенным модулем (**смарт-картой**). В целях обеспечения максимальной устойчивости для защиты контрольных слов **клиенты ECI** могут создавать защищенный канал связи **клиента ECI** со **смарт-картой** или (в целях безопасности) непосредственно от **смарт-карты** к блоку усовершенствованной системы безопасности. Подробные сведения о действующих протоколах по обмену данными при управлении ключами не определяются интерфейсом **ECI**, однако в полной мере определяются системой CA/DRM на основе API блока **усовершенствованной системы безопасности**, как указывается в [ITU-T J.1014].

ECI-совместимые устройства **CPE** могут иметь один или несколько слотов для считывания карт. **Хост ECI** управляет устройствами считывания карт, причем процесс управления полностью прозрачен для **клиентов ECI**. **Хост ECI** сопоставляет каждую вставляемую **смарт-карту** с доступными **клиентами ECI**. С этой целью **клиенты ECI** передают список спецификаторов карт **хосту ECI**. **Хост ECI** разрешает возможные конфликты между **клиентами ECI**, которые намерены получить доступ к одной и той же **смарт-карте**. **Хост ECI** обеспечивает также урегулирование конфликтов между устройствами считывания карт.

9.5.3.2 Основные спецификации

В настоящем разделе приведены основные стандарты и спецификации, которым должны соответствовать устройства для считывания карт оборудования **CPE**, а также соответствующие драйверы и программное обеспечение **хоста ECI**.

Физические характеристики устройства для считывания карт оборудования **CPE** могут быть основаны на соответствующих требованиях рынка. Преобладающим для карт условного доступа является формат ID-1 (размер кредитной карты); используются также карты формата ID-000 (SIM). Для справки см. [ISO/IEC 7816-1], [ISO/IEC 7816-2] и [ISO/IEC 14496-12].

Типовое устройство для считывания карт оборудования **CPE** должно поддерживать как минимум операции класса A (5B) и B (3B) в соответствии с разделом 5 [ISO/IEC 7816-3]. Должны поддерживаться следующие контакты: C1 (VCC), C2 (RST), C3 (CLK), C5 (GND) и C7 (I/O).

Хосты ECI могут поддерживать устройства для считывания карт, не соответствующие указанным выше требованиям. Такие устройства должны иметь специальную маркировку, для того чтобы **пользователь** не принял их за типовые устройства для считывания карт с поддержкой **ECI**.

Хост ECI и устройства для считывания карт оборудования **CPE** должны поддерживать соответствующие функции **ECI**, определяемые в разделах 6–12 [ISO/IEC 7816-2]. **Хост ECI** инициирует любую вставляемую карту с использованием процедур, определяемых в [ISO/IEC 7816-2].

Хост ECI должен использовать функциональные возможности согласно [ISO/IEC 7816-3], необходимые для реализации технических характеристик, приведенных в настоящей Рекомендации. **Хост ECI** должен по мере необходимости соблюдать требования [ISO/IEC 7816-5] для поддержки функциональных возможностей системы поиска AID, определяемых в пункте 9.5.3.3 ниже.

9.5.3.3 Управление доступом к смарт-картам

Перед инициированием соединения с **клиентом ECI хост ECI** инициирует протокол и устройство для считывания карт в соответствии с разделами 6–11 [ISO/IEC 7816-3]. Хост выбирает соответствующие настройки протокола, параметры синхронизации связи и операционный класс **смарт-карты**.

Хост ECI должен быть способен получить AID (идентификатор приложения, определяемый в пункте 8.2.1.2 [ISO/IEC 7816-4] и пункте 8.2.1 [ISO/IEC 7816-4]. Идентификатор AID извлекается из карты, как указывается в пункте 8.2.2.1 [ISO/IEC 7816-4] из архивных байтов или из строки исходных данных. Для **смарт-карт**, работающих с несколькими приложениями, **хост ECI** должен быть способен получить список идентификаторов AID, как указывается в пункте 8.2 [ISO/IEC 7816-4], а именно в пунктах 8.2.1.1, 8.2.2 и 8.2.2.3.

Хост ECI использует следующий список идентификаторов карты:

1) Если карта работает с несколькими приложениями, то в соответствии с [ISO/IEC 7816-4] она должна использовать в качестве списка идентификаторов карты список идентификаторов AID, извлеченный из шаблонов приложений EF.DIR, а также использовать идентификаторы AID, непосредственно представленные в EF.DIR.
2) Если карта не работает с несколькими приложениями, то в соответствии с вышеуказанным пунктом 1) идентификатор AID, извлеченный из "архивных байтов", как определяется в пунктах 8.1.1 или 8.1.2 [ISO/IEC 7816-4], должен использоваться как единственный идентификатор карты.
3) Если идентификатор AID не может быть извлечен, как определяется в вышеуказанных пунктах 1) и 2), то в качестве единственного идентификатора карты должен использоваться ATR, определяемый в пункте 8.2 [ISO/IEC 7816-4]. В целях сопоставления ATR определяется значениями от T0 до Tk, исключая TCK (при наличии).

На основании вышеуказанного списка идентификаторов карт оборудование **CPE** выполняет сопоставление с **клиентами ECI**.

Клиенты ECI предоставляют список приемлемых спецификаторов для идентификаторов карт, если он готов для подключения к карте. Атрибут эксклюзивной карты присутствует в каждом спецификаторе идентификатора карты и указывает, что **хост ECI** должен отправить **пользователю** информацию о разрешении конфликта доступа к **смарт-карте**. Это справедливо для случая, когда несколько **клиентов ECI** запрашивают доступ к **смарт-карте**, соответствующей спецификатору идентификатора карты, и такая **смарт-карта** вставляется или присутствует в одном из устройств для чтения **смарт-карт** оборудования **CPE**.

Хост ECI обнаруживает и по возможности разрешает конфликты между идентификацией карты и сопоставлением **клиентов ECI** в соответствии со следующими правилами.

- **Смарт-карта** считается соответствующей **клиенту ECI**, если один из идентификаторов карты в списке идентификаторов соответствует одному из спецификаторов идентификатора карты **клиента ECI**.

- В случае если **смарт-карта** соответствует нескольким **клиентам ECI** и ни один из **клиентов ECI** не претендует на эксклюзивный доступ, сеанс карты предоставляется в следующем порядке:
  - сначала организуется сеанс карты для клиента ECI, у которого был последний сеанс работы с картой;
  - если такого клиента ECI не существует или карта не распознается устройством для считывания карт оборудования **CPE**, сеанс работы с картой может быть организован по алгоритму, выбираемому **хостом ECI**.
- Клиент **ECI** отключает сеанс работы со **смарт-картой**, если не может взаимодействовать со **смарт-картой**, чтобы **хост ECI** мог сопоставить ее с другими **клиентами ECI**, которые могут пытаться использовать ее.

Клиенты ECI должны иметь возможность обрабатывать генерируемые **хостом ECI** события "подключения" и "отключения" в сеансе работы со **смарт-картой**.

9.5.3.4 Разрешение конфликтов устройств для считывания карт

В настоящем пункте определяются функциональные возможности приложений **хостов ECI** по разрешению конфликтов между клиентами и имеющимися устройствами для считывания карт для обеспечения доступа к **смарт-картам**.

Клиент ECI обеспечивает приоритет сеанса работы со **смарт-картой** при доступе к **смарт-картам** через устройство для считывания карт (сеанс **смарт-карты**). Используются следующие значения:

- **Active** (активный) – используется для основной функции, прерывание которой создает неудобства для **пользователя**. Примером может служить сеанс просмотра, запрашиваемый **пользователем**, или сеанс записи, предварительно запрограммированный **пользователем**;
- **Background** (фоновый) – используется для фоновой обработки данных, которая при необходимости может быть прервана (состояние по умолчанию). Примером может служить обработка сообщений EMM для получения прав доступа в будущем.

Клиент ECI должен быть способен запрашивать введение смарт-карты (подразумевая, что она будет активно использоваться), ссылаясь на один или несколько **указателей медиаданных** или строку, указывающую, какое приложение требует карту, в случае если карту не требует конкретный **указатель медиаданных**.

Если **клиент ECI** запрашивает карту, **хост ECI** направляет **пользователя** к соответствующему устройству для считывания карт, используя следующие рекомендации:

- предпринимаются попытки направить пользователя к доступному устройству для считывания карт;
- предпринимаются попытки направить пользователя к устройству считывания, работающему в фоновом режиме, если нет свободного устройства;
- если фоновый режим или свободные устройства для считывания не доступны, предпринимаются попытки направить пользователя к устройству для считывания, работающему в активном режиме, что причиняет наименьшие неудобства **пользователю**. При этом используется информация приложения/**клиента ECI** о действующих активных сеансах с этими устройствами.

В указанном выше процессе может участвовать **хост ECI**, использующий дополнительную информацию для сопоставления карты типу устройства для считывания (например, физические размеры). Например, чтобы связать тип устройства для считывания с **клиентом ECI**, который соответствует техническим требованиям для подключения к **клиенту ECI** при условии, что тот же тип карты будет использоваться в будущем. С этой целью **хост ECI** может использовать собственный набор правил.

9.5.3.5 API управления сеансом работы со смарт-картой

9.5.3.5.1 Общие сведения

API управления сеансом работы со **смарт-картой** предоставляет клиентам управляемый доступ к **смарт-картам**, как определяется в пунктах 9.5.3.3 и 9.5.3.4.

Доступные сообщения API для управления сеансом работы со **смарт-картой** перечисляются в таблице 9.5.3.5.1-1.

Таблица 9.5.3.5.1-1 – Сообщения API управления сеансом работы со смарт-картой

Сообщение	Тип	Направление	Маркер	Описание
setCardMatch	set	C→H	0x0	Установка списка спецификаторов идентификации карты для **клиента ECI**
callCardSessionPrio	call	C→H	0x1	Установка приоритета сеанса работы со **смарт-картой**
getCardConnStatus	get	H→C	0x2	Предоставляет статус состояния при подключении посредством карты
reqCardConOpen	A	H→C	0x3	Сообщает **клиенту ECI**, что сеанс работы с картой открыт
reqCardConClose	A	H→C	0x4	Сообщает **клиенту ECI**, что сеанс работы с картой закрыт
reqCardConClose	A	C→H	0x5	Сообщает **хосту ECI**, что **клиент ECI** намерен завершить сеанс работы с подключенной картой

9.5.3.5.2 Сообщение setCardMatch

C→H setCardMatch(uint **matchListLenth,** CardSpecifier **matchList[])**

- Это сообщение позволяет **клиенту ECI** указывать идентификаторы карт, с помощью которых он намерен подключаться.

Определение свойства CardMatch

matchListLength: uint	Длина matchList в отношении спецификаторов
matchList: CardSpecifier[].	См. таблицу 9.5.3.6.1-1. "Сообщения API связи с использованием **смарт-карты**". **Хост ECI** использует этот список для сопоставления подключенных **смарт-карт** с **клиентом ECI** в соответствии с пунктом 9.5.3.3. Определение типа приводится в таблице 9.5.3.5.2-1, а значения для поля specifierType указывается в таблице 9.5.3.5.2-2

Таблица 9.5.3.5.2-1 – Определения типа для спецификатора смарт-карты

```
#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;
```

Таблица 9.5.3.5.2-2 – Тип спецификатора смарт-карты

Название	Значение	Описание
CardSpecifierATR	0x01	Спецификатор карты типа ATR. Карта соответствует спецификатору, если поле **atrLen** равно длине ATR карты, а байты ATR карты соответствуют первым байтам **atrLen** поля **atr**. ATR карты определяется в пункте 9.5.3.5.3, T0..TCK
CardSpecifierAID	0x02	Спецификатор карты типа AID. Карта соответствует спецификатору, если поле **aidLen** равно длине идентификатора AID карты, а байты AID карты соответствуют первым байтам **aidLen** поля **aid**. AID карты определяется в пункте 9.5.3.3
RFU	Прочее	Зарезервировано для использования в будущем

Предварительные условия

1) **Клиент ECI** готов отправить отклик на сообщения invCardConOpen и invCardConClose, если **matchListLength** > 0.

Постусловия

1) **Хост ECI** сопоставляет с **клиентом ECI** любую карту, вставленную в устройство для считывания карт, как определяется в пункте 9.5.3.3. В случае совпадения хост открывает при необходимости сеанс работы посредством карты с **клиентом ECI**, как определяется в пункте 9.5.3.5.5.

2) **Хост ECI** не прекращает текущий сеанс работы карты в том случае, если новый список matchList не совпадает с подключенной в данный момент **смарт-картой**. Для этой цели **клиент ECI** использует сообщение reqCardConnClose.

9.5.3.5.3 Сообщение callCardSessionPrio

C→H callCardSessionPrio(uchar **priority,** uint **nrMh,** ushort **mH[],** char ***clientApplication)**

- Это сообщение обновляет приоритет сеанса посредством карты и предоставляет х**осту ECI** список mH **указателя медиаданных** и внутреннюю причину **клиента ECI** для отправки запроса или проведения активного сеанса с картой.

Определения параметра вызова

priority: uchar	Приоритет сеанса работы с картой, требуемый **клиентом ECI**. Значения определяются в таблице 9.5.3.5.3-1
nrMh: uint	Количество **указателей медиаданных** в зависимости от активного сеанса работы с картой
mH: ushort	Список **указателей медиаданных**, для которых требуется активный сеанс работы со **смарт-картой**
clientApplication: char *	Символ null завершает строку с указанием причины для **клиента ECI**, по которой требуется активный сеанс работы со **смарт-картой**, не связанный с **указателем медиаданных**. Если этот указатель равен NULL, то такого требования не существует. Если указатель не равен NULL, то значение строки для пользователя должно быть отличным от нуля. Максимальное количество отображаемых символов составляет 40

Таблица 9.5.3.5.3-1 – Значения приоритета сеанса работы со смарт-картой

Название	Значение	Описание
CardPriorityBackground	0x01	Запрос приоритета карты **клиентом ECI** выполняется в фоновом режиме. Определение запроса дается в пункте 9.5.3.4
CardPriorityActive	0x02	Запрос приоритета карты **клиентом ECI** выполняется в активном режиме. Определение запроса дается в пункте 9.5.3.4
RFU	Прочее	Зарезервировано для использования в будущем

Постусловия

1) **Хост ECI** управляет сеансом работы с картой, как определяется в пункте 9.5.3.4, в соответствии с **приоритетом** и использует параметры **mH** и **clientApplication** для разрешения конфликтов доступа к устройствам для считывания карт через интерфейс **пользователя**, если это необходимо.

9.5.3.5.4 Сообщение getCardConnStatus

C→H uchar **getCardConStatus()**

- Это сообщение возвращает статус текущего сеанса подключения к **смарт-карте**.

Определение свойства: см. таблицу 9.5.3.5.4-1.

Таблица 9.5.3.5.4-1 – Значения статуса подключения карты

Название	Значение	Описание
CardConNo	0x00	**Клиент ECI** не проводит сеанс работы со **смарт-картой**
CardConYes	0x01	**Клиент ECI** проводит сеанс работы со **смарт-картой**
RFU	Прочее	Зарезервировано для использования в будущем

9.5.3.5.5 Сообщение reqCCardConOpen

H→C reqCCardConOpen() →
C→H resCardConOpen()

- Это сообщение позволяет **хосту ECI** информировать **клиента ECI** о новом событии – сеансе подключения с помощью карты; **клиент ECI** отправляет отклик, подтверждающий обработку события.

Предварительные условия (запрос)

1) Сеанс работы карты с **клиентом ECI** должен быть установлен в соответствии с пунктом 9.5.3.3.

Постусловия (отклик)

1) **Клиент ECI** управляет приоритетом сеанса в соответствии с требованиями, указанными в пункте 9.5.3.4.
2) **Клиент ECI** закрывает сеанс при отсутствии необходимости работы с картой, как определяется в пункте 9.5.3.3.

9.5.3.5.6 Сообщение reqCCardConClose

H→C reqCCardConClose () →
C→H resCardConClose ()

- Это сообщение позволяет **хосту ECI** информировать **клиента ECI**, что сеанс работы с картой закрыт. **Клиент ECI** в своем отклике подтверждает, что событие обработано.

Предварительные условия (запрос)

1) Карта удалена из устройства для считывания или произошел серьезный сбой в работе функционирования подсистемы устройства для считывания карт, который вызвал потерю соединения.

Постусловия (отклик)

1) **Отклик клиента ECI** подтверждает, что **клиент ECI** обработал событие и готов принять новое соединение с помощью карты, как определяется свойством CardMatch.

9.5.3.5.7 Сообщение reqHCardConClose

C→H reqHCardConClose() →
H→C reqHCardConClose ()

- Это сообщение позволяет **клиенту ECI** информировать **хост ECI** о том, что у него нет необходимости во взаимодействии с подключенной **смарт-картой**.

Постусловия (отклик)

1) **Хост ECI** подключает **смарт-карту** к другому соответствующему **клиенту ECI**, как указывается в пункте 9.5.3.3, и не предпринимает попыток подключить эту карту к **клиенту ECI** (отложенные перезагрузки и циклы включения/выключения питания).
2) **Хост ECI** дожидается получения **отклика** перед возможным повторным подключением другой соответствующей **смарт-карты** к **клиенту ECI**.

9.5.3.6 Определения сообщений API связи с использованием смарт-карты

9.5.3.6.1 Общие сведения

API **откликов**/команд управления **смарт-картой** предоставляет примитивы сеанса связи между **клиентом ECI** и **смарт-картой** в контексте открытого сеанса **смарт-карты**, управляемого **хостом ECI**. **Клиент ECI** может выполнять обмен командами/откликами [ISO/IEC 7816-3] с **хостом ECI** на уровне блоков APDU (см. примечание), как указывается в разделе 12 [ISO/IEC 7816-3]. **Клиент ECI** имеет доступ ко всем функциям управления **смарт-картой** и может выполнять сброс и повторное инициирование с пользовательскими настройками параметров, если это необходимо, а также получать данные о параметрах связи. Сообщения API ECI определяются в таблице 9.5.3.6.1-1.

ПРИМЕЧАНИЕ. – Возможен также обмен протоколами T=0 на уровне TPDU посредством обмена короткими командами и откликами на интерфейсе уровня APDU.

Таблица 9.5.3.6.1-1 – Сообщения API связи с использованием смарт-карты

Сообщение	Тип	Направление	Маркер	Описание
reqCardCmdRes	A	C→H	0x6	Отправка команды при помощи карты, получение отклика при помощи карты
reqCardReInit	A	C→H	0x7	Сброс карты ("теплый" или "холодный") и повтор последовательности инициирования с установкой последнего приоритета инициирования
callCardSetProp	set	H→C	0x8	Установка параметра связи при помощи карты
callCardGetProp	get	H→C	0x9	Получение свойства/параметра связи при помощи карты

9.5.3.6.2 Сообщение reqCardCmdRes

C→H reqCardCmdRes(byte **nodeAddrByte,** uint **cmdApduLen,** byte **cmdApdu[]) →**
H→C resCardCmdRes(uint **resApduLen,** byte **resApdu[])**

- Как определено в разделе 12 [ISO/IEC 7816-3], это сообщение отправляет командный код APDU для **смарт-карты** через **хост ECI** и принимает обратно APDU отклика. Соответствующие коды ошибок определяются в таблице 9.5.3.6.2-1.

Определение параметра запроса

nodeAddrByte: byte	Байт адреса узла для параметра T=1 установленного протокола **смарт-карты**, как определяется в пункте 11.3.2.1 [ISO/IEC 7810]. Этот параметр игнорируется, если параметр протокола **смарт-карты** равен T=0
cmdApduLen: unit	Длина cmd APDU в байтах. Следует отметить, что длина внутреннего кода **cmdApdu** не должна превышать **cmdApduLen**
cmdApdu: byte []	Командный код APDU для отправки на карту. Дополнительные байты в поле cmdApdu игнорируются **хостом ECI**

Определение параметра отклика

resApduLen: uint	Длина APDU **отклика** в байтах
resAdpu: byte []	Пакет APDU **отклика**, полученный от карты

Предварительные условия (запрос)

1) **Клиент ECI** проводит открытый сеанс работы со **смарт-картой**.

2) Предыдущее сообщение reqCardCmdRes привело к resCardCmdRes или (повторно) инициировано соединение.

Таблица 9.5.3.6.2-1 – Коды ошибок resCardCmdRes

Название	Описание
ErrCardConnOpenNot	**См. таблицу** 9.5.3.7-1
ErrCardConnFail	

9.5.3.6.3 Сообщение reqCardReInit

C→H reqCardReInit(uchar **resetMode)** →
H→C resCardReInit()

- Это сообщение отправляет **хосту ECI** запрос на сброс **смарт-карты** с помощью resetMode и ее повторное инициирование с установкой последнего приоритета для соединения с помощью карты. **Отклик** возвращается после завершения процесса (или в случае ошибки). Соответствующие коды ошибок определяются в таблице 9.5.3.6.3-2.

Определение параметра запроса

resetMode: uchar	См. таблицу 9.5.3.6.3-1

Таблица 9.5.3.6.3-1 – Значения resetMode карты

Название	Значение	Описание
CardResetCold	0x01	Производится "холодный" сброс, и карта должна быть повторно инициирована, как если бы она была только что первоначально подключена и (см. пункт 6.2.3 [ISO/IEC 7816-1]
CardResetWarm	0x02	Производится "теплый" сброс; параметры синхронизации связи при помощи карты должны быть повторно инициированы (см. пункт 6.2.3 [ISO/IEC 7816-3]). Выбор протоколов и параметров, как указывается в разделе 9 [ISO/IEC 7816-3], должен быть выполнен повторно в случае необходимости. Данная процедура может использоваться, в частности, при попытке замены параметров синхронизации интерфейса на приоритетное значение для клиента ECI
RFU	Прочее	Зарезервировано для использования в будущем

Предварительные условия (запрос)

1) **Клиент ECI** проводит открытый сеанс со **смарт-картой**.

Постусловия (отклик)

1) **Отклик** указывает на успешную установку протокола интерфейса и настроек параметров.

Таблица 9.5.3.6.3-2 – Коды ошибок resCardCmdRes

Название	Описание
ErrCardConnOpenNot	См. таблицу 9.5.3.7-1
ErrCardConnFail	

9.5.3.6.4 Сообщение callCardSetProp

C→H callCardSetProp (ushort **propTag,** uin**t valueLen,** byte ***propValue)**

- Это сообщение устанавливает свойство, доступное для записи и обозначенное маркером **propTag** интерфейса **смарт-карты** для значения **propValue**.

Определение параметра запроса

propTag: ushort	Маркер свойства протокола связи при помощи карты, подлежащий изменению. Значения определены в таблице 9.5.3.6.5-2
valueLen: uint	Длина поля paramValue в байтах
propValue: byte *	Указатель для значения свойства в целях записи в параметр, обозначенный маркером propTag

Таблица 9.5.3.6.4-1 – Коды ошибок callCardSetProp

Название	Описание
ErrCardConnOpenNot	См. таблицу 9.5.3.7-1

9.5.3.6.5 Сообщение callCardGetProp

C→H callCardGetPropf(ushort **propTag,** uint **valueLen,** byte ***propValue)**

- Это сообщение считывает доступное свойство, обозначенное маркером **propTag** интерфейса **смарт-карты**, в значение **propValue**. Соответствующие коды ошибок определяются в таблице 9.5.3.6.5-1.

Определение параметра запроса

propTag: ushort	Маркер свойства протокола связи при помощи карты, подлежащий изменению. Значения определяются в таблице 9.5.3.6.5-2
valueLen: uint	Максимальная длина поля paramValue в байтах. Дополнительные байты свойства не копируются в значение propValue
propValue: byte *	Указатель на запрошенное значение свойства

Таблица 9.5.3.6.5-1 – Коды ошибок callCardSetProp

Название	Описание
ErrCardConnOpenNot	См. таблицу 9.5.3.7-1

Таблица 9.5.3.6.5-2 – Значения маркера API карт и семантика для свойств протоколов карт

Название	Значение маркера	Описание
CardPropClass	0x0001	Один байт. Значение Class A=0x01, Class B = 0x02, Class C = 0x03. Другие значения резервируются для использования в будущем. Только для чтения
CardPropAtrLen	0x0002	Один байт. Длина ATR карты в **CardPropAtr** в байтах. Только для чтения
CardPropAtr	0x0003	Строка байтов, макс. 16 байтов. ATR карты при "холодном" сбросе. Только для чтения
CardPropPpsExch	0x0004	Карта и интерфейс успешно завершили обмен PPS, если значение не равно 0x00. Только для чтения
CardPropPpsVal	0x0004	Один байт. Значение результата обмена PPS1 для PPS карты. Другие значения в настоящей Рекомендации не поддерживаются. Только для чтения
CardPropTAEff	0x0005	Один байт. Действительное значение TA, применяемое для синхронизации часов на интерфейсе. Только для чтения
CardPropTCEff	0x0006	Один байт. Действительное значение TC, применяемое для синхронизации часов на интерфейсе. Только для чтения
CardPropProt	0x0007	Один байт. Это значение указывает на протокол, выбранный устройством интерфейса для связи с помощью карты. Значения определяются в пункте 8.2.3 [ISO/IEC 7816-3], поле "T". Значение 0x00 указывает на протокол T=0, значение 0x01 указывает на протокол T=1. Могут отображаться другие значения (до 0x0E). Только для чтения
CardPropT1IFSC	0x0008	Один байт. Текущее значение протокола IFSC (размер информационного поля карты) в кодированном протоколе T=1 определяется в пункте 11.4.2 [ISO/IEC 7816-3]. Только для чтения
CardPropT1IFSD	0x0009	Один байт. Текущее значение протокола IFSD (размер информационного поля устройства = устройство для считывания карт) в кодированном протоколе T=1 определяется в пункте 11.4.2 [ISO/IEC 7816-3]. Только для чтения
CardPropAidListLen	0x000A	Один байт. Длина списка идентификаторов AID карты, полученных от карты на этапе инициирования. Только для чтения
CardPropAidList	0x000B	*(byte[MaxAid]). Список идентификаторов AID, полученных от карты на этапе инициирования. Только для чтения
CardPropClassPref	0x0011	Три байта. Последовательность приоритетных значений класса. Значения для приоритета должны быть установлены по порядку (с соблюдением безопасности). Значения трех байтов относятся к **CardPropClass**, при этом значение 0x00 означает "приоритет отменен". Чтение и запись
CardPropImplClock	0x0012	Однобайтовое значение **TA** применяется в случае, если бит 5 **TA_2** в ATR указывает на неявные значения для тактовой частоты. Чтение и запись
CardPropPps1SegLen	0x0013	Один байт. Значение является двоичным числом без знака. Минимальное значение равно 0, максимальное значение равно 0x08. Представляет количество значений PPS1 для попытки согласования обмена PPS в **CardPropPps1Seq**, как указывается в разделе 9 [ISO/IEC 7816-3]. См. примечание
CardPropPps1Seq	0x0014	Однобайтовая последовательность максимальной длиной 8 битов, начиная с наиболее желательного значения для установления PPS1 при обмене в PPS. Значения определяются в пункте 9.2 [ISO/IEC 7816-3]. Чтение и запись
CardPropInfdPref	0x0015	Один байт. Указывается предпочтительное значение IFSD, которое должно быть установлено для протокола T1 устройством интерфейса. Чтение и запись
RFU	Прочее	Зарезервировано для использования в будущем
ПРИМЕЧАНИЕ. – Значения для PPS2 и PPS3 не поддерживаются в данном APi и не требуют поддержки хостом ECI. Чтение и запись.		

9.5.3.7 Коды ошибок для API смарт-карты

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.5.3.7-1.

Таблица 9.5.3.7-1 – Коды ошибок API смарт-карты

Название	Значение	Описание
ErrCardOpenNot	–256	Сеанс работы с картой не создан
ErrCardConnFail	–257	Сеанс работы с картой создан, но соединение не установлено (после сброса)
RFU	**Прочее**	Зарезервировано для использования в будущем

9.5.4 API обнаружения карусели передачи данных

9.5.4.1 Общие сведения

API обнаружения карусели передачи данных позволяет **клиенту ECI** извлекать информацию из карусели радиовещания формата **ECI**, как указывается в пункте 7.7.2. Используя этот интерфейс, **клиент ECI** помимо прочего может получать обновленную информацию по импорту/экспорту.

ПРИМЕЧАНИЕ. –Карусели передачи данных предназначаются для передачи квазистатических данных и не являются предпочтительным транспортным протоколом для промежуточных данных.

Клиент ECI способен считывать данные непосредственно из карусели передачи данных или запрашивать **хост ECI** для контроля обновлений интересующего его модуля или группы элементов карусели. Контроль осуществляется либо в состоянии PwrOn (электропитание включено), либо в определенный промежуток времени в режиме ожидания. При этом рекомендуется (для оптимизации энергопотребления), чтобы временные промежутки совпадали с контрольными периодами **хоста ECI**.

Хост ECI предпринимает попытку получить запрашиваемые данные и сохранить их в файле, с тем чтобы в дальнейшем **клиент ECI** имел к ним доступ через API файловой системы. **Хост ECI** предоставляет минимальное количество параллельных каналов обнаружения для каждого **клиента ECI**, как предложено в [b-ITU-T J Suppl. 7].

Сообщения API обнаружения карусели передачи данных перечислены в таблице 9.5.4.1-1.

Таблица 9.5.4.1-1 – Сообщения API обнаружения карусели передачи данных ECI

Сообщение	Тип	Направление	Маркер	Описание
reqDCAcqGroupInfo	A	C→H	0x0	**Клиент ECI** посылает запрос **хосту ECI** на считывание структуры GroupInfoIndication в DSI-сообщении указанной карусели передачи данных ECI
reqDCAcqModule	A	C→H	0x1	**Клиент ECI** посылает запрос **хосту ECI** на получение определенного модуля карусели передачи данных **ECI** в файл, применяя параметры фильтра модуля и различные режимы

9.5.4.2 Сообщение reqDCAcqGroupInfo

C→H reqDCAcqGroupInfo (uint **operatorId,** uint **platformId)** →
H→C resDCAcqGroupInfo (byte **gii[])**

- **Клиент ECI** посылает запрос **хосту ECI** на считывание структуры GroupInfoIndication в DSI-сообщении указанной карусели передачи данных **ECI**. Соответствующие коды ошибок определяются в таблице 9.5.4.2-1.

Определения параметра запроса

operatorId: uint	20-битовый идентификатор **оператора**, обнаруженный в структуре ECI_carousel_id, которая содержится в дескрипторе data_broadcast_id_descriptor() в PSI (см. пункт 7.7.2.4)
platformId: uint	20-битовый идентификатор **системы управления платформой**, обнаруженный в структуре ECI_carousel_id, которая содержится в дескрипторе data_broadcast_id_descriptor () в PSI (см. пункт 7.7.2.4)

Определения параметра отклика

gii: byte[]	Байтовый массив, содержащий структуру GroupInfoIndication, который переносится в DSI карусели передачи данных, как определяется для DVB DSM-CC в [ETSI EN 301 192]

Подробная семантика

- **Хост ECI** предоставляет доступ к только к каруселям передачи данных загружаемых клиентов.

Таблица 9.5.4.2-1 – Коды ошибок reqDCGroupInfo

Название	Описание
ErrDCAcqNetwAccessResource	См. таблицу 9.5.4.4-1
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	

9.5.4.3 Сообщение 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()

- Это сообщение позволяет **клиенту ECI** посылать запрос **хосту ECI** на получение определенного модуля карусели передачи данных **ECI** в файл, применяя параметры фильтра модуля и различные режимы.

Определения параметра запроса

aid: uchar	Номер фильтра обнаружения. Максимальное количество активных фильтров обнаружения, которые может иметь **клиент ECI**, равно трем (значения 0 .. 2).
fname: fileName	Имя файла, в который должны быть скопированы данные из модуля карусели, подлежащего обнаружению. Все существующие данные перезаписываются
oId: uint	20-битовый идентификатор **оператора**, обнаруженный в структуре ECI_carousel_id, которая содержится в дескрипторе data_broadcast_id_descriptor() в PSI (см. пункт 7.7.2.4)
pId: uint	20-битовый идентификатор **системы управления платформой**, обнаруженный в структуре ECI_carousel_id, которая содержится в дескрипторе data_broadcast_id_descriptor () в PSI (см. пункт 7.7.2.4)
dType: byte	Это поле должно соответствовать полю Descriptor type группы модуля, как указывается в таблице 7.7.2.4-1
model: uint	16-битовое значение без знака, которое должно соответствовать полю model в дескрипторе compatibilityDescriptor группы, подлежащей обнаружению. См. таблицу 7.7.2.4-1
version: uint	16-битовое значение без знака, которое соответствует (положительный фильтр), не соответствует (отрицательный фильтр) или игнорируется в части соответствия полю version в дескрипторе compatibilityDescriptor группы, подлежащей получению, в зависимости от параметра **mode**, биты 0 и 1. см. таблицу 7.7.2.4-1
index: uint	Индекс модуля, к которому должен быть обеспечен доступ в группе. Этот параметр интерпретируется в соответствии с параметром **mode**, бит 1
mode: uint	Параметр состоит из нескольких полей: **бит 0 –** сообщает о положительной или отрицательной фильтрации по полю **version**: 0b0 – положительная фильтрация, 0b1 – отрицательная фильтрация; **бит 1–** сообщает, если фильтрация по полю **version** должна быть проигнорирована (значение 0b1) или не проигнорирована (значение 0b0); **бит 2 –** сообщает, если индекс должен быть проигнорирован (значение 1), а также если какой-либо модуль должен быть получен (для одномодульных каруселей) или должен ли использоваться индекс (modulo numberOfModules, см. таблицу 7.7.2.6-1); **бит 29 –** если такой бит задан, то **хост ECI** выполняет обнаружение в режиме ожидания, проверяя карусель на соответствие своим собственным требованиям по обнаружению; подобное обнаружение должно продолжаться до тех пор, пока не будет получено уведомление о получении запрашиваемых данных как в режиме ожидания, так и в режиме powerOn; **бит 30 –** сообщает, что в процессе обнаружения предполагается наличие запущенной карусели передачи данных, а обнаружение должно быть завершено в сроки, установленные расписанием (значение 0b0), либо сообщает о том, должно ли обнаружение продолжаться, если карусель может быть обнаружена и если фильтр обнаружения соответствует значению (0b1) (то есть необходимо дождаться, пока не появятся данные); **бит 31 –** включает (значение 0b1) или отключает (значение 0b0) обнаружение с помощью указанного фильтра **aid**

Предварительные условия (отклик)

1) **Запрашиваемый** модуль карусели был обнаружен, произошла ошибка файловой системы или если возникла проблема обнаружения при установке бита 30 параметра **mode.**

2) **Хост ECI** находится в состоянии PwerOn; то есть **клиент ECI** не активируется при обнаружении в режиме ожидания.

Постусловия (отклик)

1) Этот файл содержит указанный модуль или возникла ошибка.

2) Если установлен бит 30 параметра **mode**, ошибки обнаружения появляться не должны.

Подробная семантика

- **Хост ECI** предоставляет доступ только к каруселям данных загружаемых **клиентов ECI**, для которых осуществляется контроль карусели передачи данных в целях, установленных **хостом ECI**.
- Если параметр не задан, то обнаружение в режиме ожидания не выполняется. Для создания собственного расписания обнаружения **клиенты ECI** могут использовать интерфейс Wakeup API, описываемый в пункте 9.4.7.3.
- **Хост ECI** предоставляет "тривиальный" **отклик** в случае запроса при сброшенном бите 31 параметра **mode**.

Соответствующие коды ошибок перечисляются в таблице 9.5.4.3-1.

Таблица 9.5.4.3-1 – Коды ошибок reqDCAcqModule

Название	Описание
ErrDCAcqNetwAccessResource	См. таблицу 9.5.4.4-1
ErrDCAcqNetwAccessFail	
ErrDCAcqNoCarousel	
ErrDCAcqCarNoGroup	
ErrDCAcqCarNoModule	
ErrDCAcqCarTimeout	
ErrDCAcqFileSystemFailure	
ErrDCAcqFileQuotaExceeded	

9.5.4.4 Коды ошибок для API обнаружения карусели данных

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.5.4.4-1.

Таблица 9.5.4.4-1 – Коды ошибок API сеанса для медиаданных транспортных потоков

Название	Значение	Описание
ErrDCAcqNetwAccessResource	–256	См. таблицу 9.6.2.3.7-1
ErrDCAcqNetwAccessFail	–257	См. таблицу 9.6.2.3.7-1
ErrDCAcqNoCarousel	–258	В радиовещательных сетях, доступных для **хоста ECI**, не обнаружено каруселей с соответствующим идентификатором **оператора** и **системы управления платформой**
ErrDCAcqCarNoGroup	–260	Структура groupInfoIndication обнаружена в DSI карусели, однако соответствующая группа не найдена
ErrDCAcqCarNoModule	–261	Группа карусели (DII) обнаружена, но соответствующий модуль не найден
ErrDCAcqCarTimeout	–262	Произошла задержка в доступе к DSI, DII или DDB карусели
ErrDCAcqFileSystemFailure	–263	См. таблицу 9.4.5.5-1
ErrDCAcqFileQuotaExceeded	–264	См. таблицу 9.4.5.5-1

9.6 Интерфейсы API для доступа к ресурсу дешифрования хоста ECI

9.6.1 API дешифрования хоста ECI

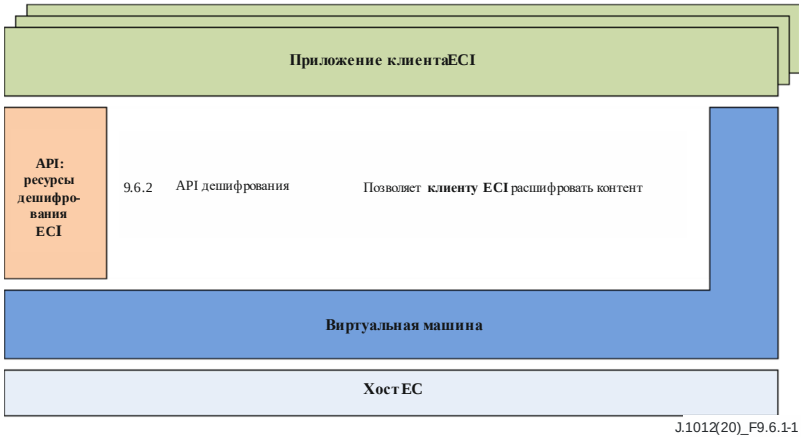


Рисунок 9.6.1-1 – Блок-схема интерфейсов API, определяемых в пункте 9.6

В таблице 9.6.1-1 перечисляются интерфейсы API, рассматриваемые в пункте 9.6, а рисунок 9.7.1-1 иллюстрирует расположение интерфейсов API, определяемых в пункте 9.6, а также архитектуру **ECI**.

Таблица 9.6.1-1 – Список интерфейсов API, определяемых в пункте 9.6

Пункт	Наименование API	Описание
9.6.2	API дешифрования **хоста ECI**	Позволяет **клиенту ECI** передавать **хосту ECI** информацию о стандартном URI, относящуюся к определенному элементу контента

9.6.2 Определение API дешифрования хоста ECI

9.6.2.1 Введение

Интерфейсы API дешифрования позволяют **хосту ECI** (например, по запросу резидентных или загружаемых приложений) выбирать **клиента ECI**, соответствующего требованиям к дешифрованию контента, и запрашивать его на предмет дешифрования. Обмен всеми сообщениями дешифрования между **клиентом ECI** и **хостом ECI** производится в контексте **указателя медиаданных**, представляющего контент, любую соответствующую сеть доставки и ресурсы, необходимые для его дешифрования.

Следующие интерфейсы API образуют API дешифрования:

1) общий API сеансов передачи медиаданных для всех типов мультимедиа, включающий функцию сопоставления контента и **клиента ECI**;
2) интерфейсы API дешифрования транспортных потоков;
3) интерфейсы API дешифрования файлов и потоков APIs.

9.6.2.2 API сеансов передачи медиаданных

9.6.2.2.1 Общие сведения

Клиент ECI может сообщать список спецификаторов соответствия, посредством которых **хост ECI** может сопоставлять его с контентом.

Хост ECI может посылать запрос соответствующему **клиенту ECI** на открытие сеанса дескремблирования для **указателя медиаданных**. Открытие сеанса не означает, что будет запущено декодирование. Оно лишь гарантирует, что ресурсы, необходимые для доступа к контенту и/или метаданным в нем и для проведения сеанса дескремблирования, доступны как со стороны **хоста ECI**, так и со стороны **клиента ECI**. **Клиенты ECI** должны обеспечивать доступ к **смарт-картам** и другим ресурсам, необходимым для фактического дескремблирования контента, и их готовность до начала сеанса. В таблице 9.6.2.2.1-1 перечисляются функции API.

Таблица 9.6.2.2.1-1 – Сообщения API сеанса дешифрования указателя медиаданных

Сообщение	Тип	Направление	Маркер	Описание
setDcrMhMatch	Set	C→H	0x0	Сообщает **хосту ECI**, по каким идентификаторам может быть опознан **клиент ECI** для дескремблирования контента
reqDcrMhOpen	A	H→C	0x1	**Хост ECI посылает клиенту ECI запрос** на открытие специального сеанса передачи медиаданных с использованием **указателя медиаданных**
reqDcrMhClose	A	H→C	0x2	**Хост ECI** закрывает сеанс передачи медиаданных с **клиентом ECI**
reqDcrMhBcAlloc	A	C→H	0x3	**Клиент ECI запрашивает** сеанс **указателя медиаданных** для целей доступа к своей собственной радиовещательной сети
reqDcrMhCancel	A	C→H	0x4	**Клиент ECI** отменяет сеанс передачи медиаданных с **хостом ECI**

9.6.2.2.2 Сообщение API setDcrMhMatch

C→H setDcrMhMatch(uint **matchListLength,** MatchSpecifier **matchList[])**

- Это сообщение разрешает **клиенту ECI** передать **хосту ECI** идентификаторы системы дешифрования, для которой могут быть предоставлены услуги дешифрования транспортного потока.

ПРИМЕЧАНИЕ. – Реальная возможность дешифрования контента может зависеть от подписки, статуса оплаты и других факторов.

Определение свойства SetDcrMhMatch

matchListLength: uint	Длина списка matchList в отношении спецификаторов
matchList: MatchSpecifier[].	Таблица 9.6.2.2.2-1. **Хост ECI** использует данный список для сопоставления контента с потенциальными возможностями **клиента ECI** по дешифрованию согласно пункту 9.5.3.3. Спецификаторы соответствия определяются типом MatchSpecifier. Для получения соответствия все поля MatchSpecifier должны соответствовать контенту

Таблица 9.6.2.2.2-1 – Определения типа для MatchSpecifier

```
#define MaxMhSubFormat  16;
typedef struct MatchSpecifier {
    uchar decryptIdType; /*см. таблицу 9.6.2.2.2-2 */
    union decryptId {
        bool        ECI Client ID;
        ushort  dvbCaId;
        byte        uuid[16];
    }
    byte mhType;
    byte subFormat[MaxMhSubFormat];
} MatchSpecifier;
```

Таблица 9.6.2.2.2-2 – Определение setDcrMhMatch decryptIdType

Название	Значение	Описание
None	0x00	Не соответствует контенту по созданному **запросу**; при открытии сеанса отображает *no match* (несоответствие)
ClientEciId	0x01	Идентификация **клиента ECI** может быть выполнена на основе идентификатора **клиента ECI**, составленного из 20-битовых значений (поля типа и версии не включены) <<operator_id,platform_operation_id>,<vendor_id,client_id>>, определяемых в разделе 7 настоящей Рекомендации
ClientDvbCaId	0x02	decryptId является идентификатором системы условного доступа, определяемым в [CEN EN 50221] и [ETSI EN 301 192]. Данное значение указывает на то, что dvbCaId используется как вариант объединения specifierType. Значения actual+ для dvbCaId определяются в [CEN EN 50221]
ClientUUID	0x03	Дешифрован идентификатор DRM ID (как это определено параметром CENC/Dash), обозначаемый как UUID [IETF RFC 4122]
RFU	Прочее	Зарезервировано для использования в будущем

mhType: unit	Тип **указателя медиаданных** (основной режим дешифрования), поддерживаемого **клиентом ECI** для данного ClientEciID
subFormat: byte[]	Данный параметр позволяет определить дополнительную спецификацию типа для **клиента ECI**. Интерпретация этих байтов зависит от параметра **mhType**, определяемого в таблице 9.6.2.2.2-3

Таблица 9.6.2.2.2-3 – Определение типа subFormat

Значение mhType	Семантика поля subFormat
ISOBMFF	Поле subFormat содержит от нуля и более последовательных определений 4CC маркированных значений блоков ftyp или styp ISOBMF, подходящих для декодирования **клиентом ECI**. Одно (или несколько) из этих значений 4CC должны соответствовать значениям major_brand или compatible_brands блока ftyp или styp контейнера ISOBMFF. Значение, равное 0x0000, в subFormat указывает на отсутствие значения (постоянное несоответствие); значение, равное 0xFFFF, в качестве первой записи может соответствовать любому маркированному значению (независимо от последующих байтов)
Прочее	Зарезервировано для использования в будущем

Подробная семантика

Предпринимая попытки рендеринга контента на основе транспортного потока, **хост ECI** должен пытаться сопоставлять контент с доступными **клиентами ECI**, используя следующие правила в порядке приоритета:

1) **Хост ECI** должен пытаться сформировать набор применимых спецификаторов соответствия на основе идентификаторов **клиентов ECI** для данного контента, как указывается в пункте 7.2.2. Если какой-либо применимый идентификатор **клиента ECI** и связанные с ним свойства соответствуют спецификатору **MatchSpecifier** одного **клиента ECI**, этому **клиенту ECI** предоставляется контент для дешифрования. Если соответствуют несколько **клиентов ECI**, **хост ECI** использует следующую процедуру:

   a) **хост ECI** предоставляет контент для дешифрования тому **клиенту ECI**, который осуществил последнюю успешную передачу контрольного слова для дешифрования контента из того же самого источника контента;

   b) Если первый **клиент ECI** не способен дешифровать контент, он должен пытаться использовать соответствующих альтернативных **клиентов ECI**; при этом **клиенты ECI** должны использоваться в порядке, соответствующем последним успешным случаям дешифрования, связанным с источником контента.

2) Если **хост ECI** не может задать какой-либо идентификатор **клиента ECI** для контента или если ни один из **клиентов ECI** согласно пункту 1), указанному выше, не способен декодировать контент, **хост ECI** должен пытаться задать набор других идентификаторов для контента, как указывается в пункте 9.5.4.3. Если только один идентификатор и связанные с ним свойства соответствуют одному **клиенту ECI**, **хост ECI** предоставляет **клиенту ECI** контент для дешифрования. Если соответствуют несколько **клиентов ECI**, **хост ECI** использует следующую процедуру:

a) **хост ECI** предоставляет контент для дешифрования тому **клиенту ECI**, который осуществил последнее успешное дешифрование контента из того же самого источника контента;

b) если первый **клиент ECI** не способен дешифровать контент, он должен пытаться использовать соответствующих альтернативных **клиентов ECI**; при этом **клиенты ECI** должны использоваться в порядке, соответствующем последним успешным фактам дешифрования, связанным с источником контента.

Термин "источник контента", приведенный выше, должен охватывать как минимум:

1) вещательную сеть DVB или пакет программ, при этом являющиеся источником транспортного потока;

2) веб-сайт, используемый для просмотра ресурсов сети, а также браузер, предоставляющий ссылки на контент.

9.6.2.2.3 Сообщение reqDcrMhOpen

H→C reqDcrMhOpen(ushort **mH,** MatchSpecifier **match) →**
C→H resDcrMhOpen(ushort **mH)**

- Это сообщение позволяет **хосту ECI** запрашивать сеанс дешифрования у **клиента ECI. Клиент ECI** должен резервировать все ресурсы, в большинстве случаев необходимые для выполнения дешифрования, как определяется параметрами **mh** и **match**. Соответствующие коды ошибок указываются в таблице 9.6.2.2.3-1.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** контента, подлежащего дешифрованию
match: MatchSpecifier	Копия спецификатора соответствия (содержит также тип указателей медиаданных для сеанса)

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего дешифрованию

Предварительные условия (запрос)

- **Хост ECI** зарезервировал все ресурсы, необходимые для дешифрования контента. Если речь идет о контенте транспортного потока, то в него входят ресурсы настройки или доступа к другой сети, а также применимые ресурсы управления, ресурсы демультиплексирования и ресурсы дескремблирования для как минимум одного приложения cw-pair.

Постусловия (отклик)

- При положительном результате **клиент ECI** резервирует все ресурсы, необходимые в большинстве случаев для декодирования контента в запрашиваемом сеансе. Должен также быть включен доступ к любым внешним ресурсам (серверы DRM, **смарт-карты** и т. д.), которые обычно требуются для выполнения дешифрования.

ПРИМЕЧАНИЕ. – Ресурсы, требуемые в виде исключения, или ресурсы, доступные при нормальных условиях по мере необходимости, исключаются.

- При возврате ErrDcrUserDelay **клиент ECI** ожидает ввода данных **пользователем** для открытия сеанса (например, для получения доступа к **смарт-карте**). **Хост ECI** должен повторять отправку запроса reqDcrMhOpen (с такими же параметрами) до возврата либо положительного результата, либо точно определенной ошибки; как вариант, он может отправлять запрос reqDcrMhClose для завершения ожидающего сеанса. **Клиент ECI** может отменять запрос reqDcrMhCancel в том случае, если он не может получить необходимые данные, введенные **пользователем**.

Таблица 9.6.2.2.3-1 – Коды ошибок reqDcrMhOpen

Название	Описание
ErrDcrUserDelay	См. таблицу 9.6.2.2.7-1
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.2.4 Сообщение reqDcrMhClose

H→C reqDcrMhClose(ushort **mH) →**
C→H resDcrMhClose(ushort **mH)**

- Это сообщение позволяет **хосту ECI** закрыть сеанс дешифрования с **клиентом ECI. Клиент ECI** может освободить ресурсы для данного сеанса.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** сеанса, подлежащего закрытию

Определение параметра отклика

mH: ushort	**Указатель медиаданных** закрытого сеанса

Постусловия (запрос)

- **Клиент ECI** освобождает любые ресурсы, которые он затребовал специально для сеанса.

Постусловия (отклик)

- **Хост ECI** может освободить любые ресурсы, относящиеся к **указателю медиаданных**.

9.6.2.2.5 Сообщение reqDcrMhBcAlloc message

C→H reqDcrMhBcAlloc(byte **networkType[2],** uchar **priority,** char **reason[80]) →**
H→C resDcrMhBcAlloc(ushort **mH)**

- Это сообщение позволяет **клиенту ECI** запрашивать подключение к радиовещательной сети в целях сбора данных по безопасности.

Определение параметра запроса

networkType: byte[2]	Тип радиовещательной сети, доступ к которой предоставляется клиенту ECI; значения соответствуют таблице 9.6.2.3.6.2-3
priority: uchar	Приоритет доступа к сети определяется в таблице 9.6.2.2.5-1
reason: char[80]	Строка максимальной длиной 80 символов, завершающаяся нулевым символом, которая может быть предоставлена **пользователю** в целях разрешения конфликта ресурсов в хосте ECI при выполнении данного запроса

Таблица 9.6.2.2.5-1 – Определение приоритета доступа к радиовещательной сети

Название	Значение	Описание
DcrAllocPrioBackground	0x01	Доступ необходим для фоновой обработки, который может быть не предоставлен или может быть прерван, если доступа к ресурсам требует задача с более высоким приоритетом. Примером может служить доступ к EMM или к защищенным данным восстановления на центральном мультиплексе
DcrAllocPrioActivec	0x02	Доступ необходим для первичной функции дескремблирования. Если доступ не предоставляется (или прерывается), это создает дискомфорт для **пользователя**. Примером может служить сеанс просмотра, запрашиваемый **пользователем**, или сеанс записи, предварительно запрограммированный **пользователем**
RFU	Прочее	Зарезервировано для использования в будущем

Определение параметра запроса

mH: ushort	**Указатель медиаданных** открытого сеанса

Подробная семантика

- Используя сообщение reqDcrMhClose, **хост ECI** может отменить сеанс в том случае, если для другой задачи требуются ресурсы доступа к сети с более высоким приоритетом.

- **Клиент ECI** закрывает сеанс, используя сообщение reqDcrMhCancel, если ему более не требуется доступ к сети.

Постусловия (запрос)

1) **Хост ECI** выделил все ресурсы для доступа к сети запрашиваемого типа.

Постусловия (отклик)

1) **Клиент ECI** перестраивается на получение транспортного потока, используя сообщение reqDcrTsRelocate перед началом получения сегмента.

Таблица 9.6.2.2.5-2 – Коды ошибок reqDcrMhBcAlloc

Название	Описание
ErrDcrNetworkAccessCapability	См. таблицу 9.6.2.2.7-1
ErrDcrNetworkAccessResource	
ErrDcrPrioOverride	
ErrDcrResourceMissing	

9.6.2.2.6 Сообщение reqDcrMhCancel

C→H reqDcrMhCancel(ushort **mH,** uchar **reason) →**
H→C resDcrMhCancel(ushort **mH)**

- Это сообщение позволяет **клиенту ECI** закрывать сеанс дешифрования с **хостом ECI**. **Клиент ECI** освободил все ресурсы, специально требуемые для сеанса.

Определения параметра запроса

mH: ushort	**Указатель медиаданных** сеанса, подлежащего закрытию
reason: uchar	Причина отмены сеанса дешифрования. Значения определяются в таблице 9.6.2.2.6-1

Таблица 9.6.2.2.6-1 – Значения для обоснования отмены reqDcrMhCancel

Название	Значение	Описание
DrcMhUndefined	0x00	Произошла неизвестная ошибка у **клиента ECI**, вынуждающая его отменить сеанс
DcrMhCardMissing	0x01	**Смарт-карта**, требуемая для декодирования, не может быть успешно подключена (повторно подключена) и использована для дешифрования контента в течение разумно необходимого периода времени
DcrMhServiceMissing	0x02	Служба (не относящаяся к **CPE**), поддерживающая предоставление **клиентом ECI** услуг дешифрования, требуемых для проведения сеанса дешифрования, недоступна в течение разумно необходимого периода времени
DcrMhResourceMissing	0x03	Ресурс, относящийся к **CPE** и требуемый для предоставления услуг дешифрования, недоступен для клиента **ECI** в течение в течение разумно необходимого периода времени (исключая **DcrMhMmiMissing**)
DcrMhMmiMissing	0x04	**Клиенту ECI** не удалось получить ресурс сеанса MMI для взаимодействия с **пользователем**, требуемого для проведения сеанса дешифрования в течение разумно необходимого периода времени
DcrMhAllocTerminate	0x05	**Указатель медиаданных**, выделенный для **клиента ECI** посредством reqDcrMhBcAlloc, более не требуется **клиенту ECI**
RFU	Прочее	Зарезервировано для использования в будущем

Разумно необходимый период времени для отмены сеанса **указателя медиаданных хостом ECI** предлагается в [b-ITU-T J Suppl. 7].

Определение параметра отклика

mH: ushort	**Указатель медиаданных** отмененного сеанса

Предварительные условия (запрос)

- **Клиент ECI** освобождает ресурсы, которые ему требовались специально для сеанса.

Постусловия (запрос)

- **Хост ECI** может освободить любые ресурсы, относящиеся к **указателю медиаданных**.

Постусловия (отклик)

- Сеанс указателя медиаданных закрывается хостом ECI.

9.6.2.2.7 Коды ошибок для API сеанса передачи медиаданных

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.6.2.2.7-1.

Таблица 9.6.2.2.7-1 – Коды ошибок API сеанса передачи для медиаданных транспортных потоков (TS)

Название	Значение	Описание
ErrDcrUserDelay	–256	Произошла длительная задержка при ожидании ввода данных **пользователем,** необходимых для завершения операции. Операция не завершена
ErrDcrCardMissing	–257	**Смарт-карта**, требуемая для сеанса, недоступна или не готова
ErrDcrServiceMissing	–258	Услуга, получаемая извне (то есть не относящаяся к оборудованию **CPE**) и необходимая для выполнения **клиентом ECI** операций дешифрования, недоступна
ErrDcrResourceMissing	–259	Неизвестный ресурс в составе оборудования **CPE**, необходимый для доступа или дешифрования контента, недоступен
ErrDcrMmiMissing	–260	Доступ **клиента ECI** к интерфейсу MMI отсутствует
ErrDcrDescrContinue	–261	**Хост ECI** продолжает попытки дескремблирования контента в данном TS
ErrDcrNetworkAccessCapability	–262	**Хост ECI** не обладает ресурсом доступа к сети для определения расположения запрошенного TS.
ErrDcrNetworkAccessResource	–263	**Хост ECI** не может получить ресурс доступа к сети для доступа к запрашиваемому TS
ErrDcrPrioOverride	–264	Задача с более высоким приоритетом в **CPE** требует ресурсов для **указателя медиаданных**, что приводит к прекращению сеанса **указателя медиаданных**
RFU	Прочее	Зарезервировано для использования в будущем

9.6.2.3 Дескремблирование данных транспортного потока

9.6.2.3.1 Введение

Хост ECI может посылать **клиенту ECI** запрос на проведение сеанса дескремблирования (сеанс особого типа – в данном случае радиовещание в формате mpeg), предоставляя ему **указатель медиаданных** (см. пункт 9.1.2). **Хост ECI** предоставляет данные системы безопасности, определяемые **клиентом ECI**, для дескремблирования данных.

Для дескремблирования контента в большинстве форматов транспортного потока интерфейс **ECI** использует скрытую модель для синхронизации контрольных слов с контентом, предоставляемым дескремблеру. В этой модели **хост ECI** предоставляет **клиенту ECI** контрольные данные системы безопасности из транспортного потока по мере его демультиплексирования и дескремблирования. **Клиент ECI** предоставляет требуемые контрольные слова (обычно два слова на каждый элементарный поток, зачастую одинаковые для всех элементарных потоков) в надлежащий интервал времени. Как правило, **клиент ECI** декодирует сообщения ECM и преобразует их в контрольные слова, которые незамедлительно загружаются в дескремблер. Применение этих контрольных слов синхронизируется с потоком посредством сигнализации в потоке контента с использованием битов управления скремблированием на уровне TS-пакетов или на уровне PES-пакетов.

Интерфейс API разделяется на следующие пункты:

1) запуск, перезапуск и остановка дешифрования транспортного потока (пункт 9.6.2.3);
2) сбор данных по безопасности (пункт 9.6.2.3.5);
3) функции настройки радиовещания (пункт 9.6.2.3.6).

9.6.2.3.2 Версии формата и сеанса транспортного потока

Транспортные потоки, которые дескремблируются посредством **указателя медиаданных** в сеансе передачи медиаданных типа **MhDvbTsBroadcast**, должны соответствовать следующим спецификациям: [ISO/IEC 13818-1-1] (в частности, приложение битов управления скремблированием к TS-пакетам) и [ETSI ETR 289].

9.6.2.3.3 Требования по обработке хоста ECI

9.6.2.3.3.1 Обнаружение шифра скремблирования

Хост ECI передает **клиенту ECI** информацию о применимом режиме шифрования, используя следующие правила.

1) Хост ECI использует для потоков DVB сигнализацию с применением дескриптора скремблирования в PMT, как определяется в [ETSI TS 103 127] и [ETSI TS 100 289].

2) Если дескриптор не обнаружен по ссылке 1) и источником является радиовещательная сеть DVB, то **хост ECI** предполагает, что применяется CSA1, как указывается в определении дескриптора скремблирования.

9.6.2.3.3.2 Обнаружение идентификации условного доступа

Для формирования списка применимых идентификаторов условного доступа DVB для скремблированной услуги, если скремблирование обнаруживается транспортным потоком или битами скремблирования пакетов PES, в транспортном потоке (создаваемым от радиовещательной сети или иного источника) **хост ECI** использует следующую последовательность правил обнаружения:

1) Хост ECI предпринимает попытку извлечь CA_descriptors из таблицы PMT услуги.

2) Если извлечь дескрипторы не удается, и контент заскремблирован, хост ECI предпринимает опытку извлечь набор идентификаторов CA_system_id, содержащихся в дескрипторе идентификатора CA пакета программ DVB, таблиц SDT или EIT, применимых для контента.

ПРИМЕЧАНИЕ. – Для некоторых источников контента на основе транспортного потока применимый идентификатор CA или DRM можно определить, используя другие средства.

9.6.2.3.4 Запуск и остановка дешифрования транспортного потока

9.6.2.3.4.1 Общие сведения

Хост ECI способен запускать дешифрование контента на базе открытого **указателя медиаданных**, используя резервные ресурсы **клиента ECI**. **Хост ECI** предоставляет таблицу CA-PMT, которая содержит спецификацию элементарных потоков, подлежащих дешифрованию. В таблице 9.6.2.3.4.1-1 перечисляются доступные сообщения API дешифрования.

Таблица 9.6.2.3.4.1-1 – API дешифрования контента транспортного потока указателя медиаданных

Сообщение	Тип	Направление	Маркер	Описание
reqDcrTsDescrStart	A	H→C	0x08	**Запросы клиенту ECI** на дескремблирование или возврат статуса дескремблирования программы в транспортном потоке
reqDcrTsDescrStop	A	H→C	0x09	**Хост ECI** посылает **клиенту ECI** запросы на дескремблирование **указателя медиаданных**
reqDcrTsDescrQuit	A	C→H	0x0A	**Клиент ECI** завершает сеанс дескремблирования с **хостом ECI**

9.6.2.3.4.2 Сообщение reqDcrTsDescrStart

H→C reqDcrTsDescrStart(ushort **mH,** uint **caPmtLen,** byte **caPmt[]) →**
C→H resDcrTsDescrStart(ushort **mH,** unit **sizeofEsStat,** descrStat **esStat[])**

- Это сообщение позволяет **клиенту ECI** начать дешифрование программы, определяемой параметром **caPmt**, в потоке, который идентифицируется значением **mH**, либо запросы о наличии возможности или условий для данной операции.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
caPmtLen: uint	Длина параметра **caPmt** в байтах
caPmt: byte[]	Объект ca_pmt определяется в пункте 8.4.3 [ETSI TR 101 202] в сетевом порядке байтов совместно с измененной интерпретацией параметров ca_pmt_list_management и ca_pmt_cmd_id, как указывается в таблице 9.6.2.3.4.2-1

Значения параметра ca_pmt_list_management и семантика должны соответствовать определениям, приведенным в таблице 9.6.2.3.4.2-1.

Таблица 9.6.2.3.4.2-1 – Значения ca_pmt_list_management

Название	Значение	Описание
DcrTsDescrStartOnly	0x03	Отдельная программа подлежит дескремблированию в рамках услуги. Это значение может быть новым или обновленным
DcrTsDescrStartUpdate	0x05	Означает то же, что и **DcrTsDescrStartOnly**
RFU	Прочее	Зарезервировано для использования в будущем

Значения параметра ca_pmt_cmd_id должны быть идентичны значениям, приведенным в пункте 8.4.3 [CEN EN 50221], с учетом следующих ограничений.

1) Значение 0x02 (ok_mmi) не допускается.

2) Значения 0x01 (ok_scrambling) и 0x03 (query) не должны встречаться в одной и той же структуре ca_pmt. Иными словами, **запрос** должен представлять собой либо простой запрос, либо простой запрос дескремблирования

Определения параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока
sizeofEsStat: uint	Количество байтов в параметре esStat
esStat: descrStat	Статус дескремблирования элементарных потоков, указываемый в параметре **caPmt запроса**. Значение descrStat определяется в таблице 9.6.2.3.4.2-2. Значение **descrStat.pid** встречается в **esStat** лишь один раз. Если соответствующий параметр ca_pmd_cmd_id не равен 0x04 (not_selected), то каждый параметр elementary_PID структуры ca_pmt [CEN EN 50221] встречается один раз. Если ca_pmd_cmd_id равен 0x04 (not_selected), elementary_PID не должен встречаться в esStat

Таблица 9.6.2.3.4.2-2 – Определение типа для структуры descrStat

```
typedef struct descrStat {
    ushort pid;
    uchar      caStatus
} descrStat;
```

pid: ushort	Значение PID потока, подлежащего дескремблированию
caStatus: uchar	Значения должны соответствовать определению параметра CA_enable объекта ca_pmt_reply в пункте 8.4.3 [CEN EN 50221]

Подробная семантика

1) **Хост ECI** выдает эту команду, если набор элементарных потоков, подлежащих декодированию, должен измениться.

2) **Хост ECI** направляет **запрос reqDcrTsDescrEnd**, если сеанс передачи данных остановлен. Ошибка при выполнении данной операции может ввести **клиента ECI** в заблуждение, и он начинает регистрацию потребляемого **пользователем** контента и начисление соответствующей платы.

3) Соответствующие коды ошибок определяются в таблице 9.6.2.3.4.2-3.

Предварительные условия (запрос)

1) **mH** открыт и представлен в формате транспортного потока.

Постусловия (запрос)

2) **Клиент ECI** может запустить процесс дескремблирования и использовать другие относящиеся к транспортному потоку функции mH.

Таблица 9.6.2.3.4.2-3 – Коды ошибок reqDcrTsStart

Название	Описание
ErrDcrUserDelay	См. таблицу 9.6.2.3.7-1
ErrDcrCardMissing	
ErrDcrServiceMissing	
ErrDcrResourceMissing	
ErrDcrMmiMissing	

9.6.2.3.4.3 Сообщение reqDcrTsDescrStop

H→C reqDcrTsDescrStop(ushort **mH) →**
C→H resDcrDescrStop(ushort **mH)**

- Это сообщение позволяет **хосту ECI** указывать **клиенту ECI** на необходимость остановки процесса дескремблирования транспортного потока, связанного с текущим параметром **mH**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока

Определение параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока

Предварительные условия (отклик)

1) Любые действия клиента **ECI**, связанные с дескремблированием mH, прекращаются.

9.6.2.3.4.4 Сообщение reqDcrTsDescrQuit

C→H reqDcrTsDescrQuit(ushort **mH, ushort** reason**) →**
H→C resDcrDescrQuit(ushort **mH)**

- Это сообщение позволяет **клиенту ECI** информировать **хост ECI** об остановке в целях обработки ключей для процесса дескремблирования транспортного потока, связанного с текущим параметром **mH**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
reason: ushort	Основания, по которым **клиент ECI** завершает обработку ключей для процесса дескремблирования, как определено в таблице 9.7.2.5.9-1

Определение параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока

Предварительные условия (отклик)

1) Все действия **хоста ECI**, связанные с дескремблированием **mH**, прекращаются или возвращается ошибка.

Постусловия (отклик)

2) Все действия **клиента ECI**, связанные с **mH**, немедленно прекращаются или возвращается ошибка.

Таблица 9.6.2.3.4.4-1 – Коды ошибок reqDcrTsDescrQuit

Название	Описание
ErrDcrDescrContinue	См. таблицу 9.6.2.3.7-1

9.6.2.3.5 Обнаружение данных дешифрования клиента ECI в транспортном потоке

9.6.2.3.5.1 Общие сведения

Клиент ECI может выполнять во внутриполосном транспортном потоке сбор данных, необходимых для дешифрования, в форме секций транспортного потока, связанных с **указателем медиаданных**. Наиболее прямой формой является установка фильтра секций. Чтобы ускорить процесс обнаружения при смене канала может быть установлен по умолчанию фильтр секций, включая таблицу PMT и поток ECM. Могут также считываться стандартные таблицы MPEG и DVB из **хоста ECI**. Секции MPEG являются структурами данных, как указывается в пункте 2.4.4.1 [ISO/IEC 13818-1-1], структура private_section(). Функции данной части API TS MPEG перечисляются в таблице 9.6.2.3.5.1-1.

Таблица 9.6.2.3.5.1-1 – Сообщения управления дескремблированием TS хоста ECI

Сообщение	Тип	Направление	Маркер	Описание
setDcrTsSectionAcqDefault	set	C→H	0x10	Устанавливает фильтр по умолчанию для обнаружения секций
setDcrTsSectionAcq	set	C→H	0x11	Устанавливает фильтр для обнаружения секций
reqDcrTsSection	A	H→C	0x12	Переадресует обнаруженную секцию **клиенту ECI**
reqDcrTsTable	A	C→H	0x13	**Клиент ECI** обнаруживает таблицу в потоке

9.6.2.3.5.2 Спецификация фильтров секций

Секции MPEG, как указано в пункте 2.4.4.11 [ISO/IEC 13818-1-1], могут извлекаться согласно спецификации из транспортного потока и передаваться от **клиента ECI хосту ECI. Хост ECI** должен поддерживать восемь фильтров секций для **клиента ECI**. Настройки фильтра секций позволяют **клиенту ECI** осуществлять фильтрацию одного идентификатора PID в транспортном потоке с ограниченным количеством косвенных спецификаторов (например, для PMT). Настройки позволяют **клиенту ECI** устанавливать положительные фильтры (где выбранные поля секций соответствуют спецификации **клиента ECI**) и отрицательные фильтры (данные секции отличаются от спецификации фильтра **клиента ECI**). Отфильтрованные секции могут объединяться в группы и передаваться при достижении максимального размера буфера либо переадресовываться по мере получения.

При фильтрации байтов секции второй и третий байты пропускаются.

Спецификация для фильтра секций приводится в таблице 9.6.2.3.5.2-1.

Таблица 9.6.2.3.5.2-1 – Определение типа для 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;
```

Семантика

pid: ushort	PID TS-пакетов, подлежащих фильтрации. Значения PID представляются 13-битовым значением без знака: то есть в диапазоне от 0x0000 до 0x1FFF. PID таблицы PMT потока, подлежащего обнаружению, представляется значением 0x8000. PID для соответствующего потока ECM, подлежащего обнаружению, представляется значением 0x8001
caId: ushort	Данное поле применимо только, если значение поля **pid** равно 0x8001. В данном случае значением этого поля является идентификатор CA MPEG/DVB системы условного доступа, для которой должен быть обнаружен поток ECM. **Хост ECI** анализирует таблицу PMT услуги, подлежащей дескремблированию, и сопоставляет поле **caId** с дескрипторами CA_descriptors (как указывается в [ISO/IEC 13818-1-1]), применимыми к PID видео (при наличии) или с первым элементарным потоком в таблице PMT, и использует поле CA-PID в подходящем дескрипторе для идентификации потока ECM, подлежащего обнаружению и фильтрации
bufferSize: ushort	Максимальный размер буфера. Как минимум, одна секция должна быть буферизована. Если значение этого поля задано равным нулю, каждая секция будет переадресовываться отдельно
timeout: uint	Время ожидания в мс для фильтрации отдельной секции. Перезапускается при успешной фильтрации каждой секции. Нулевое значение указывает на отсутствие времени ожидания
modeFlags: uint	После того как задан бит 0, **хост ECI** должен предотвращать отправку одной и той же секции **клиенту ECI** дважды. С этой целью **хост ECI** использует буфер предварительно обнаруженных секций с максимальным размером 64 кбайта. Другие биты зарезервированы и должны быть заданы равными 0 **клиентом ECI**
filter: byte []	Значение для сопоставления с соответствующими байтами секций
mask: byte[]	Если значение бита задано равным нулю, соответствующее согласование со значением секции игнорируется
neg: byte []	Если значение бита задано равным единице, соответствующее согласование с битом секции отрицательное

Секция согласуется с фильтром, если все положительно отфильтрованные биты секции с маской согласуются с соответствующими значениями фильтра и ни один отрицательно отфильтрованный бит секции с маской секции не соответствует своему значению фильтра (при условии существования как минимум одного отрицательно отфильтрованного бита). Соответствие секции (представленное **данными** для байтов секций 1 и 3–18) определяется функцией **sectionFilterMatch**.

```
bool sectionFilterMatch(byte *data, *filter, *mask, *neg) {
    int i;
    bool posMatch, negMatch;

    posMatch = True;
    negMatch = True;

    /* если все отрицательные байты равны 0, для отрицательного фильтра всегда обеспечивается
соответствие */
    для (i=0; i< DcrSectionFilterMaxlen; i++)
        negMatch &&= neg[i] == 0;

    /* соответствие данных секции критериям положительной и отрицательной фильтрации*/
    для (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 Сообщение reqDcrTsSectionAcqDefault

C→H setDcrTsSectionAcqDefault(ushort **mH,** uchar **filterNr,** dcrSectionFilterSpec **sectionFilter)**

- Это сообщение устанавливает фильтры секций по умолчанию, которые будут использоваться **хостом ECI** для получения информации из потока для **клиента ECI** после получения сообщения resDcrTsDescrStart. Данная функция может, например, использоваться **клиентом ECI** для того, чтобы ускорить обнаружение секции для сообщений ECM **хостом ECI** в процессе смены канала.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока, на котором устанавливается фильтр секций по умолчанию
filterNr: uchar	Номер программируемого фильтра. Это значение находится в диапазоне от 0 до 7
sectionFilter: dcrSectionFilterSpec	Спецификация фильтра секций согласно пункту 9.6.2.3.5.2 dcrSectionFilterSpec

Постусловие

- Этот фильтр секции вводится в действие **хостом ECI** непосредственно после успешного получения **resDcrTsDescrStart**. **Хост ECI** должен рассчитывать на успешное получение **resDcrTsDescrStart**, если для этого есть достаточно оснований.

9.6.2.3.5.4 Сообщение reqDcrTsSectionAcq

C→H setDcrTsSectionAcq(ushort **mH,** uchar **filterNr,** dcrSectionFilterSpec **sectionFilter**

- Это сообщение устанавливает фильтры секций, которые будут использоваться **хостом ECI** для получения информации из потока **mH** для **клиента ECI**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока, на котором устанавливается фильтр секций по умолчанию
filterNr: uchar	Номер программируемого фильтра. Это значение находится в диапазоне от 0 до 7
sectionFilter: dcrSectionFilterSpec	Спецификация фильтра секций согласно пункту 9.6.2.3.5.2 dcrSectionFilterSpec

Подробная семантика

- Использование данного сообщения после установки фильтра секций по умолчанию модифицирует фильтр секций до создания следующего сообщения resDcrTsDescrStart на том же **указателе медиаданных**, который выполняет сброс фильтра секций до состояния по умолчанию (если установлены параметры по умолчанию).

Набор постусловий

- Фильтр секций вводится в действие **хостом ECI**.

9.6.2.3.5.5 Сообщение reqDcrTsSection

H→C reqDcrTsSection(ushort **mH,** uchar **filterNr,** uint **sectionDataLen**, byte **sectionData[]) →**
C→H resDcrTsSectionAcq (ushort **mH,** uchar **filterNr)**

- Это сообщение отправляет **клиенту ECI** одну или несколько секций, обнаруженных **хостом ECI** в контексте транспортного потока, идентифицируемого показателем **mH**, и фильтра, идентифицируемого показателем **filterNr**.
- Соответствующие коды ошибок определяются в таблице 9.6.2.3.5.5-1.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока, на котором устанавливается фильтр секций по умолчанию
filterNr: uchar	Номер программируемого фильтра. Это значение находится в диапазоне от 0 до 7
sectionDataLen: uint	Количество байтов в **sectionData**
sectionData: byte []	Последовательность private_sections (байтов в сетевом порядке) определяется в пункте 2.4.4.11 [ISO/IEC 13818-1-1]. Секция, содержащая ошибку CRC, не передается **клиенту ECI**

Определение параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока
filterNr: uchar	Номер запрограммированного фильтра

Предварительные условия (запрос)

1) Секции должны быть обнаружены **хостом ECI** в соответствии со спецификацией фильтра секций или время ожидания для фильтра истекло.

2) Предыдущее сообщение **reqDcrTsSection** подтверждено параметром **resDcrTsSection**.

Постусловия (отклик)

1) Следующее сообщение **reqDcrTsSection от** того же фильтра может быть отправлено **хостом ECI**.

Таблица 9.6.2.3.5.5-1 – Коды ошибок reqDcrTsSection

Название	Описание
ErrDcrTsSectionTimeout	См. таблицу 9.6.2.3.7-1
ErrDcrTsSectionCrcErr	

9.6.2.3.5.6 Сообщение reqDcrTsTable

C→H reqDcrTsTable(ushort **mH,** uchar **tableId,** uint **timeout,** uint **maxLen)**
H→C resDcrTsTable(ushort **mH,** uint **tableDataLen,** byte **tableData[])**

- Это сообщение запрашивает **хост ECI** на предмет отправки секций, формирующих стандартную таблицу или субтаблицу, применимую к программе, дескремблированной на **mH**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока, на котором устанавливается фильтр секций по умолчанию
tableId: uchar	Номер программируемого фильтра. Действительные значения перечисляются в таблице 9.6.2.3.5.6-1
timeout: uint	Время ожидания в миллисекундах. Нулевое значение указывает на отсутствие времени ожидания
maxLen: uint	Максимальное количество возвращаемых байтов sectionData. **Хост ECI** округляет количество секций до наибольшего количества в пределах заданного лимита

Таблица 9.6.2.3.5.6-1 – Значения ca_pmt_list_management

Название	Значение	Описание
DcrTsTableMpegPat	0x0000	Таблица PAT согласно [ISO/IEC 13818-1-1]
DcrTsTableMpegCat	0x0001	Таблица CAT согласно [ISO/IEC 13818-1-1]
DcrTsTableMpegPmt	0x0002	Таблица PMT выбранной программы в соответствии с [ISO/IEC 13818-1-1]. Результат пустой, если приложение использует составную таблицу PMT
DcrTsTableDvbNit	0x0140	Таблица NIT фактической сети доставки, как определено в [ETSI EN 300 468] и [ETSI TS 101 211]. В кабельных сетях, использующих NIT_{other} для передачи таблиц, связанных с регионами размещения сетей, должна указываться применимая таблица NIT_{other} для региона размещения оборудования CPE
DcrTsTableDvbSdt	0x0142	Таблица SDTactual_current, определяемая в [ETSI EN 300 468] и [ETSI TS 101 211]
DcrTsTableDvbBat	0x014A	Таблица BATactual, описанная в [ETSI EN 300 468] для пакета программ, активно используемого **хостом ECI** и/или его приложением
DcrTsTableDvbEitPf	0x014E	Текущая и следующая таблицы EITactual, определяемые в [ETSI EN 300 468] и [ETSI TS 101 211]
DcrTsDescrStartUpdate	0x05	Означает то же, что и **DcrTsDescrStartOnly**

Определение параметра отклика

mH: ushort	**Указатель медиаданных транспортного потока**
tableDataLen: uint	Количество байтов в tableData
tableData: byte []	Последовательность private_sections (байтов в сетевом порядке), представляющая (суб)таблицу, определяется в пункте 2.4.4.11 [ISO/IEC 13818-1-1]

Подробная семантика

- **Хост ECI** использует фильтры секций для сбора недавно полученных данных по всем таблицам, которые могут быть запрошены **клиентом ECI** (а также для решения других задач). Секции таблиц отправляются **хостом ECI** один раз. **Хост ECI** приостанавливает отклик, если существует необходимость получения запрашиваемой таблицы. Таблица должна быть обновлена и содержать последние полные данные, доступные для **хоста ECI**. Коды ошибок определяются в таблице 9.6.2.3.5.6-2.

ПРИМЕЧАНИЕ. – В будущем таблица может быть в любое время заменена следующей версией в потоке.

– Минимальные значения частоты повторения для обновления соответствующих таблиц DVB SI предлагаются в [b-ITU-T J Suppl. 7].

– PAT, CAT, и PMT: данные получены более чем 20 секунд назад.

Таблица 9.6.2.3.5.6-2 – Коды ошибок reqDcrTsTable

Название	Описание
ErrDcrTsSectionTimeout	См. таблицу 9.6.2.3.7-1
ErrDcrTsSectionCrcErr	

9.6.2.3.6 Управление источником клиента ECI

9.6.2.3.6.1 Общие сведения

Клиент ECI способен считывать тип источника транспортного потока, управлять (перенаправлять) источником транспортного потока, а также перенаправлять программы и/или компоненты, которые декодируются **хостом ECI**. Конкретные сообщения перечисляются в таблице 9.6.2.3.6.1-1.

Таблица 9.6.2.3.6.1-1 – Сообщения API управления источником клиента TS

Сообщение	Тип	Направление	Маркер	Описание
getDcrTsSource	get	C→H	0x18	**Клиент ECI** получает источник транспортного потока
reqDcrTsRelocate	A	C→H	0x19	**Клиент ECI** перемещает источник транспортного потока
reqDcrTsSelectPrg	A	C→H	0x1A	**Клиент ECI** выбирает по номеру программу в транспортном потоке
reqDcrTsSelectPmt	A	C→H	0x1B	**Клиент ECI** выбирает программу в транспортном потоке с помощью PMT
reqDcrTsSelectCancel	A	C→H	0x1C	**Клиент ECI** отменяет выбранную ранее программу

9.6.2.3.6.2 Сообщение getDcrTsSource

C→H tsSourceType **getDcrTsSource(**ushort **mH**)

- Это сообщение возвращает тип источника **указателя медиаданных** исходя из типа сети и указателя местоположения в сети.

Определение параметра

mH: ushort	**Указатель медиаданных** транспортного потока для получения типа и расположения настраиваемого потока

Определение свойства

Определения свойств приводятся в таблице 9.6.2.3.6.2-1.

Таблица 9.6.2.3.6.2-1 – Определение типа для структуры tsSourceType

```
#define MaxTsSourceDescr 254

typedef struct tsSourceType{
    ushort tsSourceTag ;
    byte   tsSourceDescr[MaxTsSourceDescr] ;
} tsSourceType ;
```

tsSourceTag: ushort	Тип источника TS. Определенные значения перечислены ниже, включая соответствующее значение **tsSourceDescr**
tsSourceDescr: byte[MaxTsSourceDescr]	Значение зависит от tsSourceTag, как указано в т**аблице** 9.6.2.3.6.2-2

Таблица 9.6.2.3.6.2-2 – Значения маркера tsSource

Название	Значение	Описание
tsSourceDvbTuner	0x0001	Источником транспортного потока является тюнер DVB. Параметр tsSourceDescr содержит одиночный дескриптор из таблицы 9.6.2.3.6.2-3 в сетевом порядке байтов
tsSourceDvbFile	0x0002	Источником транспортного потока является файл или другой не подлежащий настройке объект, например, IP-сеть (см. [b-ETSI TS 102 034]). Поле tsSourceDescr не определено
tsDvbDuplet	0x8003	Источник или транспортный поток может быть найден с использованием идентификатора исходной сети и идентификатора транспортного потока в рамках действующей сети. Дескриптор tsSourceDescr содержит следующий сетевой порядок байтов: struct dvbDuplet {ushort onid; ushort tsid}; Данное значение не возвращается с помощью сообщения getDcrTsSource (вместо этого сообщение возвращает tsSourceDvbTuner), однако может использоваться в сообщении reqDcrTsRelocate
RFU	Прочее	Зарезервировано для использования в будущем

Значения, превышающие 0x7FFF, не являются абсолютными указателями местоположения и не возвращаются с помощью сообщения **getDcrTsSource**.

Таблица 9.6.2.3.6.2-3 – Дескрипторы источника в виде тюнера DVB

Имя дескриптора доставки DVB	Значение маркера дескриптора DVB
terrestrial_delivery_system_descriptor	0x5A
T2_delivery_system_descriptor	0x7F, 0x04
satellite_delivery_system_descriptor	0x43
S2_delivery_system_descriptor	0x79
cable_delivery_system_descriptor	0x44
C2_delivery_system_descriptor	0x7F, 0x0D

Дескрипторы используются, как это указывается в [ETSI EN 300 468], и должны содержать одну назначенную частоту.

9.6.2.3.6.3 Сообщение reqDcrTsRelocate

C→H reqDcrTsRelocate(ushort **mH**, tsSourceType **tsLoc**) →
H→C resDcrTsRelocate(ushort **mH)**

- Это сообщение отправляет **хосту ECI** запрос на перемещение источника транспортного потока в **tsLoc**. Соответствующие коды ошибок определяются в таблице 9.6.2.3.6.3-1.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока для перемещения/перенастройки
tsLoc: tsSourceType	Новое расположение транспортного потока определяется в таблице 9.6.2.3.6.2-1

Определение параметра отклика

mH: ushort	**Указатель медиаданных** перемещенного транспортного потока

Подробная семантика

- Если требуемый ресурс доступа к сети (например, тюнер/демодулятор для радиовещания) отличается от уже выделенного ресурса согласно **указателю медиаданных**, **запрос** может быть не предоставлен **хостом ECI** в связи с ограничениями ресурса.
- При успешном завершении перенастройки все текущие процессы фильтрации и/или дескремблирования прекращаются. Как только транспортный поток обнаружен, должно быть запущено обнаружение с настройками по умолчанию.

Таблица 9.6.2.3.6.3-1 – Коды ошибок reqDcrTsRelocate

Название	Описание
ErrDcrTsNetworkAccessCapability	См. таблицу 9.6.2.3.7-1
ErrDcrTsNetworkAccessResource	
ErrDcrTsNetworkAccessFail	

9.6.2.3.6.4 Сообщение reqDcrTsSelectPrg

C→H reqDcrTsSelectPrg(ushort **mH,** ushort **prgNumber) →**
H→C resDcrTsSelectPrg(ushort **mH)**

- Это сообщение задает значение выбранной программы для дескремблирования **хостом ECI** в текущем транспортном потоке, равное **prgNumber**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
prgNumber: ushort	Номер программы в таблицах PAT и PMT MPEG (см. [ISO/IEC 13818-1-1]) в транспортном потоке, определяющем данную услугу, выбирается **хостом ECI**

Определение параметра отклика

mH: ushort	**Указатель медиаданных транспортного потока**

Подробная семантика

- **Хост ECI** определяет расположение таблицы PAT в транспортном потоке, обозначенным указателем **mH**. Он определяет расположение PID таблицы PMT путем сопоставления **prgNumber** с program_number. Хост получает таблицу PMT из обнаруженного PID и, используя стандартные функции **хоста ECI**, выбирает компоненты программы для рендеринга. При условии успешного выполнения указанных операций **хост ECI** создает **запрос reqDcrTsDescrStart для запуска** дескремблирования программы.

Постусловия (запрос)

1) Если **хост ECI** выполнял дескремблирование программы, которая не была выбрана в **запросе reqDcrTsSelectPrg** или **reqDcrTsSelectPmt**, то параметры выбора программы должны сохраняться, чтобы позже они могли быть возвращены в программу по запросу **reqDcrTsSelectCancel**.

Постусловия (отклик)

1) Если ошибка не возвращается, то далее **хост ECI** отправляет запрос **reqDcrTsDescrStart**.

Коды ошибок для этого сообщения API приводятся в таблице 9.6.2.3.6.4-1-1.

Таблица 9.6.2.3.6.4-1 – Коды ошибок reqDcrTsSelectPrg

Название	Описание
ErrDcrTsPrgNumberNotInPsi	См. таблицу 9.6.2.3.7-1
ErrDcrTsComponentSelectError	

9.6.2.3.6.5 Сообщение reqDcrTsSelectPmt

C→H reqDcrTsSelectPmt(ushort **mH,** uint **pmtLen,** byte **pmt[]) →**
H→C resDcrTsSelectPmt(ushort **mH)**

- Сообщение выбирает новую программу для дескремблирования **хостом ECI**, отправляя таблицу PMT MPEG, определяющую компоненты программы в транспортном потоке, идентифицируемом указателем **mH**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных транспортного потока**
pmtLen: uint	Количество байтов параметра **pmt**
pmt: byte	Секция private_section содержит таблицу PMT согласно [ISO/IEC 13818-1-1]

Определение параметра отклика

mH: ushort	**Указатель медиаданных транспортного потока**.

Подробная семантика

- Эта команда разрешает **клиенту ECI** выбирать в транспортном потоке компоненты, не имеющие соответствующих таблиц PAT и PMT. **Хост ECI** использует параметр **pmt** для выбора компонентов программы, подлежащих рендерингу. При условии успешного выполнения указанных операций **хост ECI** создает **запрос reqDcrTsDescrStart для запуска** дескремблирования программы.

Постусловия (запрос)

1) Если **хост ECI** выполнял дескремблирование программы, которая не была выбрана **запросом reqDcrTsSelectPrg** или **reqDcrTsSelectPmt**, то параметры выбора программы должны сохраняться, чтобы позже они могли быть возвращены в программу по запросу **reqDcrTsSelectCancel**.

Постусловия (отклик)

1) Если ошибка не возвращается, то далее **хост ECI** отправляет запрос **reqDcrTsDescrStart**.

Коды ошибок для этого сообщения API приводятся в таблице 9.6.2.3.6.5-1.

Таблица 9.6.2.3.6.5-1 – Коды ошибок reqDcrTsSelectPmt

Название	Описание
ErrDcrTsComponentSelectError	См. таблицу 9.6.2.3.7-1

9.6.2.3.6.6 Сообщение reqDcrTsSelectCancel

C→H reqDcrTsSelectCancel(ushort **mH) →**
H→C resDcrTsSelectCancel(ushort **mH)**

- Это сообщение отменяет предыдущие запросы **reqDcrTsSelectPrg** и**reqDcrTsSelectPmt**, отправленные **клиентом ECI**, возвращая их исходной программе, которая была выбрана **хостом ECI** в транспортном потоке, идентифицируемом указателем **mH**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных транспортного потока**

Определение параметра отклика

mH: ushort	**Указатель медиаданных транспортного потока**

Постусловия (отклик)

1) Далее, чтобы возобновить дескремблирование исходной программы, **хост ECI** может направить сообщение **reqDcrTsDescrStart**.

9.6.2.3.7 Коды ошибок для API сеанса передачи медиаданных транспортного потока

Значения ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются ниже в таблице 9.6.2.3.7-1.

Все запросы **указателя медиаданных**, связанные с конкретным транспортным потоком, возвращают код ошибки для параметра **указателя медиаданных** в том случае, если они применяются к **указателю медиаданных**, не связанному с TS.

Таблица 9.6.2.3.7-1 – Коды ошибок интерфейсов API сеансов передачи медиасигналов для носителей транспортных потоков

Название	Значение	Описание
ErrDcrTsUserDelay	–256	Произошла длительная задержка при ожидании ввода данных **пользователем,** необходимых для завершения операции. Операция не завершена
ErrDcrTsCardMissing	–257	**Смарт-карта**, требуемая для сеанса, недоступна или не готова
ErrDcrTsServiceMissing	–258	Услуга, не относящаяся к оборудованию **CPE** и необходимая для выполнения **клиентом ECI** операций дешифрования, недоступна
ErrDcrTsResourceMissing	–259	Неизвестный ресурс в составе оборудования **CPE**, необходимый для доступа к контенту или его дешифрования, недоступен
ErrDcrTsMmiMissing	–260	Доступ **клиента ECI** к интерфейсу MMI отсутствует
ErrDcrDescrContinue	–261	**Хост ECI** продолжает попытки дескремблирования контента в данном TS
ErrDcrTsSectionTimeout	–262	Произошла задержка при обнаружении секции
ErrDcrTsSectionCrcErr	–263	В период задержки были получены секции, но с ошибками CRC. Как правило, это означает, что данный поток сильно поврежден
ErrDcrTsNetworkAccessCapability	–264	Хост ECI не обладает ресурсом доступа к сети для определения расположения запрашиваемого TS
ErrDcrTsNetworkAccessResource	–265	Хост ECI не может обнаружить ресурс доступа к сети для доступа к запрашиваемому TS
ErrDcrTsNetworkAccessFail	–266	Ресурсу сетевого доступа к сети не удалось (достоверно) обнаружить запрашиваемый транспортный поток
ErrDcrTsPrgNumberNotInPsi	–267	Таблицу PMT с соответствующим номером программы не удалось найти исходя из таблицы PAT
ErrDcrTsComponentSelectError	–268	Не удалось выбрать компонент в PMT для демультиплексирования/дескремблирования
ErrDcrTsPidNotDescrambled	–269	Pid не выбран **хостом ECI** для дескремблирования
ErrDcrTsCwIdNotValid	–270	Ссылка на неправильный идентификатор контрольного слова
RFU	Прочее	Зарезервировано для использования в будущем

9.6.2.4 Дешифрование контента на основе файлов и потоков

9.6.2.4.1 Введение

В настоящем разделе определяется API **клиента ECI**/**хоста ECI**, позволяющий оборудованию **CPE** и загружаемым приложениям взаимодействовать с **клиентом ECI** системы безопасности посредством **хоста ECI** с целью дескремблирования контента в формате ISOBMFF [ISO/IEC 23001-9], а также любых других файлов или потоков, в которых **хост ECI** (или работающее на его основе базовое оборудование **CPE** или загружаемые приложения):

- может извлекать необходимые данные управления системой безопасности из файла или потока и передавать их клиенту ECI;
- позволяет правильно применять (синхронизировать) ключи дескремблирования, генерируемые клиентом ECI, к контенту с использованием идентификаторов Key-ID.

Файлы ISOBMFF [ISO/IEC 23001-9] представляют собой общепринятый формат упаковки файлов для множества методик загрузки, включая адаптивные и не работающие в реальном времени. Для этих форматов файлов определен также общий метод шифрования: CENC [ISO/IEC 23001-7]. Формат адаптивной потоковой передачи MPEG-Dash [ISO/IEC 23009-1] и [ETSI TS 103 285] также основан на формате ISOBMFF, а различные (в ряде случаев устаревшие) системы DRM используют собственный проприетарный субформат ISOBMFF (с коммерческим ("brand") идентификатором подписи).

Одна из секций интерфейса API разрешает **клиенту ECI** определять, какие данные ему необходимо получить из файла ISOBMFF для выполнения подобного декодирования. Таким образом, проприетарные (не соответствующие CENC) приложения DRM формата ISOBMFF могут использоваться приложениями оборудования **CPE**. Особенности дескремблирования выборок должны определяться **хостом ECI**: то есть, требуется ли соответствие CENC или наличие проприетарных расширений в **хосте ECI**.

API разделен на следующие секции:

1) Запуск и остановка дескремблирования.
2) Установка специальных фильтров сбора защищенных данных для **клиента ECI**.
3) API ключа дешифрования (контрольное слово).

9.6.2.4.2 Применимые спецификации

Файлы ISOBMFF, на которые приводятся ссылки в данном разделе, должны соответствовать [ETSI TS 103 285]. Соответствующие CENC файлы ISOBMFF (согласно требованиям к стандартному дешифрованию) должны соответствовать [ISO/IEC 23001-7].

Потоковые данные, удовлетворяющие стандарту DASH, должны соответствовать [ISO/IEC 23009-1]. **Хосты ECI**, реализующие стандарт DASH, должны (как минимум) соответствовать [ISO/IEC 23001-7], [ISO/IEC 23001-9] и [ETSI TS 103 285] в той мере, в которой эти стандарты применимы в функциональной области оборудования **CPE**.

9.6.2.4.3 Требования по обработке хоста ECI

9.6.2.4.3.1 Обнаружение идентификации системы дешифрования

Хост ECI должен быть способен получать список применимых систем дешифрования из контейнера контента согласно следующим правилам:

1) Для всех файлов ISOBMFF и MP4 **хост ECI** получает блок типа файла ('ftyp') и блок типа сегмента ('styp') и использует поле `major_brand`, а также поле `compatible_brands[]` для сопоставления контента с **клиентами ECI**.

2) Для кодированных файлов в формате ISOBMFF CENC **хост ECI** восстанавливает блоки специальных заголовков системы защиты ('pssh') из всех возможных точек расположения (см. [ISO/IEC 23001-7]) и собирает из поля `SystemID` идентификаторы UUID систем DRM, пригодных для дешифрования контента. Эти файлы могут распознаваться информационным блоком схемы защиты ('sinf'), содержащим блок типа схемы ('schm') с полем `scheme_type`, значение которого равно 'cenc' или 'cbc1', а также основной версией поля `scheme_version` с заданным значением 0x0001. Определение и расположение блоков 'sinf' указывается в [ISO/IEC 23001-7].

3) Если речь идет о контенте MPEG-Dash, **хост ECI** обнаруживает все дескрипторы ContentProtection в MPD, содержащие конкретный идентификатор UUID (начинается с "urn:uuid:xxxxx", где xxxxx – UUID) для атрибута @SchemeIdUri с целью сопоставления идентификаторов UUID DRM с **клиентом ECI,** или содержащие идентификатор системы условного доступа согласно [ETSI TS 103 285] в атрибуте @value (определение данного общего идентификатора приведено в [b-DASH-IF ID]). **Хост ECI** обнаруживает все дескрипторы ContentProtection в целях соответствия функциональным возможностям **клиента ECI**. Хост конвертирует все включенные блоки PSSH в соответствующее двоичное представление ISOBMFF.

Процесс сопоставления контента с **клиентами ECI** описывается в пункте 9.6.2.4.5.2.1.

9.6.2.4.3.2 Обнаружение типа скремблирования

Хосты ECI посылают **клиенту ECI** информацию о применимом режиме дескремблирования согласно следующим правилам:

1) Для кодированных файлов CENC в формате ISOBMFF хост должен быть способен применять правила, указанные в [ISO/IEC 23001-7] для обнаружения шифра (AES-CTR or AES-CBC), включая выбор сброшенного/скремблированного байта, заполнение, а извлечение и применение вектора инициализации определяется в [ISO/IEC 23001-7].

2) Если речь идет о контенте MPEG DASH в формате ISOBMFF, для дескремблирования применяется режим AES-CTR (с чередованием ключа), как указывается в [ETSI TS 103 285].

9.6.2.4.3.3 Фильтрация защищенных данных в контейнере контента по умолчанию

Хост ECI передает любые блоки, содержащие (непрозрачную) информацию в контейнере, предназначенном для **клиента ECI**, когда это необходимо для процесса дескремблирования. В конкретном плане это касается следующих блоков кодированных файлов в формате ISOBMFF CENC и контента Dash в формате ISOBMFF:

1) В отношении:

   a) блоков специального заголовка системы защиты в блоках 'moov' and 'moof', соответствующих идентификатору UUID идентификатора системы DRM **клиента ECI** и относящихся к контенту, который подлежит декодированию в настоящий момент или в ближайшем будущем;

   b) блоков информации схемы защиты 'sinf' – в том случае, если **клиенту ECI** требуется доступ к блокам 'sinf'.

9.6.2.4.3.4 Дескремблирование контента

Хост ECI должен отвечать за интерпретацию режима дескремблирования, идентифицируя данные, подлежащие дескремблированию, и обрабатывая данные с использованием дескремблера и соответствующих идентификаторов Key-ID для идентификации ключей, доступ к которым предоставил **клиент ECI**.

Для того чтобы **клиент ECI** мог рассчитать соответствующие ключи, **хост ECI** должен своевременно передать **клиенту ECI** необходимые данные управления безопасностью из контейнера контента.

9.6.2.4.4 API сеанса передачи медиаданных для файловых и потоковых медийных контентов

9.6.2.4.4.1 Общие сведения

Хост ECI способен запускать дешифрование контента на базе открытого **указателя медиаданных**, используя резервные ресурсы **клиента ECI**. **Хост ECI** предоставляет данные инициирования для того, чтобы **клиент ECI** начал анализ прав доступа.

Таблица 9.6.2.4.4.1-1 – API дешифрования контента транспортного потока указателя медиаданных

Сообщение	Тип	Направление	Маркер	Описание
reqDcrFileStart	A	H→C	0x01	Посылает **клиенту ECI запрос** на дескремблирование или возврат статуса дескремблирования файла или потока
reqDcrFileStop	A	H→C	0x02	**Хост ECI** посылает **клиенту ECI** запрос на остановку обработки ключей в процессе дескремблирования для **указателя медиаданных**
reqDcrFileQuit	A	C→H	0x03	**Клиент ECI** отменяет операцию дескремблирования с **хостом ECI**

9.6.2.4.4.2 Сообщение reqDcrFileStart

H→C reqDcrFileStart(ushort **mH,** uchar **reqType,** uchar **dataType,** uint **initDataLen,** byte **initData[]) → C→H resDcrFileStart(**ushort **mH,** uchar **dcrStat)**

- Это сообщение посылает **клиенту ECI** запрос на возврат статуса дескремблирования и/или запуска сеанса дескремблирования контента, связанного с mH. **Хост ECI** предоставляет **клиенту ECI** начальные данные для запуска сбора и анализа данных о лицензиях в соответствии с форматом контейнера/шифрования.

Определение параметра запроса

mH: ushort	**Указатель данных файла**
reqType: uchar	Тип **запроса** (запуск дескремблирования или запрос лицензии) определяется в таблице 9.6.2.4.4.2-1
dataType: uchar	Тип InitData
initDataLen: uint	Длина контейнера initData в байтах
initData: byte	Данные инициирования из контента, определяемые параметром dataType. Кодирование initDat определяется в таблице 9.6.2.4.4.2-2

Таблица 9.6.2.4.4.2-1 – Кодирование reqType

Название	Значение	Описание
ReqTypeDcr	0x01	Начало дескремблирование; запуск диалога с **пользователем** в случае необходимости
ReqTypeInq	0x02	Запрос опций дескремблирования
RFU	Прочее	Зарезервировано для использования в будущем

Таблица 9.6.2.4.4.2-2 – Кодирование initData

dataType	Значение	Описание
FmtIsoCenc	0x04	Обнаружено соответствие блоков PSSH ISOBMFF (см. [ISO/IEC 23001-7]) и ID DRM в спецификаторе MatchSpecifier **клиента ECI**
FmtIsoCencDash	0x05	Блоки PSSH ISOBMFF (см. [ISO/IEC 23001-7]) обнаружены в MPD (см. [ISO/IEC 23007-1]) или обнаружено соответствие сегмента инициализации (см. [ISO/IEC 23009-1]) и ID DRM в спецификаторе MatchSpecifier **клиента ECI**
FmtIsoProp	0x06	**Хост ECI** может передавать данные **клиенту ECI** на основе проприетарной информации. **Клиент ECI** должен быть способен интерпретировать данные на основе той же общей проприетарной информации
FmtIsoPropDash	0x07	В виде FmtIsoProp с указанием, что данные представляют собой источник DASH
RFU	Прочее	Зарезервировано для использования в будущем

Определение параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока
dcrStat: uchar	Статус дескремблирования; см. таблицу 9.6.2.4.4.2-3

Таблица 9.6.2.4.4.2-3 – Статус дескремблирования

Название	Значение	Описание
DcrStatNo	0x00	Дескремблирование невозможно (система DRM имеет возможность дескремблировать контент)
DcrStatOk	0x01	Начало дескремблирования; запуск диалога с **пользователем** в случае необходимости
DcrStatDialog	0x02	Требуется диалог с **пользователем**
DcrStatPay	0x03	Требуется оплата и, возможно, диалог с **пользователем**
DcrStatDrmNok	0xFE	Система DRM не имеет возможности дескремблировать этот контент
RFU	Прочее	Зарезервировано для использования в будущем

Подробная семантика

- По запросам **пользовательские** диалоги не запускаются **клиентом ECI**, однако **клиент ECI** анализирует возможность дескремблирования контента путем сброса условий лицензирования сервером лицензий без диалога с **пользователем**.

Предварительные условия (запрос)

1) Ожидание указателя медиаданных.

Предварительные условия (отклик)

1) Если клиент **ECI** способен дескремблировать контент, а параметр reqType в норме, то **клиент ECI** должен быть готов генерировать ключи дескремблирования.

Коды ошибок для сообщения-запроса запуска дешифрования приводятся в таблице 9.6.2.4.4.2-4.

Таблица 9.6.2.4.4.2-4 – Коды ошибок reqDcrFileStart

Название	Описание
ErrDcrFileUserDelay	См. таблицу 9.6.2.4.7-1
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	

9.6.2.4.4.3 Сообщение reqDcrFileStop

H→C reqDcrFile Stop(ushort **mH) →**
C→H resDcrFile Stop(ushort **mH)**

- Это сообщение позволяет **хосту ECI** останавливать процесс дешифрования.

Определение параметра запроса

mH: ushort	**Указатель данных файла**

Определение параметра отклика

mH: ushort	**Указатель данных файла**

Предварительные условия (отклик)

1) **Клиент ECI** прекратил все процессы, связанные с дешифрованием контента.

9.6.2.4.4.4 Сообщение reqDcrFileQuit

C→H reqDcrFileQuit(ushort **mH,** uint **reason) →**
H→C resDcrFile Quit(ushort **mH)**

- Это сообщение позволяет **клиенту ECI** информировать **хост ECI** о прекращении обработки ключей для процесса дешифрования файлов. Соответствующие коды ошибок определяются в таблице 9.6.2.4.4.4-1.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
reason: uint	Значения определены в таблице 9.7.2.5.9-1

Определение параметра отклика

mH: ushort	**Указатель медиаданных** файла

Предварительные условия (отклик)

1) Все действия **хоста ECI**, связанные с дескремблированием **mH**, прекращаются, или возвращается ошибка.

Постусловия (отклик)

1) Все действия **клиента ECI**, относящиеся к **mH**, немедленно прекращаются, или будет возвращена ошибка.

Таблица 9.6.2.4.4.4-1 – Коды ошибок reqDcrFileQuit

Название	Описание
ErrDcrFileDescrContinue	См. таблицу 9.6.2.4.7-1

9.6.2.4.5 Сбор данных по безопасности, связанных с клиентом ECI

9.6.2.4.5.1 Общие сведения

Хост ECI осуществляет стандартный сбор данных, которые декодируются с целью получения информации, необходимой **клиенту ECI** для расчета ключа. **Клиент ECI** может указывать и на специальный сбор данных, выходящий за рамки получения стандартных данных, предоставляемых **хостом ECI**. **Хост ECI** поддерживает ограниченное количество фильтров для сбора таких данных.

Таблица 9.6.2.4.5.1-1 – API фильтра данных

reqDcrFileFilter	req	C→H	0x04	**Клиент ECI** отправляет **хосту ECI** запрос на установку фильтра данных для сбора данных по безопасности
reqDcrFileData	A	C→H	0x05	**Клиент ECI** отправляет **хосту ECI** запрос на сбор данных через фильтр файлов

9.6.2.4.5.2 Спецификация фильтра файлов

9.6.2.4.5.2.1 Общее определение фильтра файлов

Спецификация фильтра файловых данных основана на спецификации формата файлов. Фильтр определяется в рамках контекста заданного формата файлов. Общая спецификация фильтра файлов определяется в таблице 9.6.2.4.5.2.1-1.

Таблица 9.6.2.4.5.2.1-1 – Общее определение фильтра файлов

```
typedef struct dcrFileFilterSpec {
    ushort filterType;          // определяется в таблице 9.6.2.4.5.2.1-2
    ushort filterLen;
    byte   filter[filterLen]; // должен форматироваться согласно filterType
} dcrFileFilterSpec;
```

Таблица 9.6.2.4.5.2.1-2 – Типы фильтров файлов

FileFilterIsobmff	0x0001	Фильтр файлов для данных в формате ISMBMFF определяется в пункте 9.6.2.4.5.2.2
RFU	Прочее	Зарезервировано для использования в будущем

9.6.2.4.5.2.2 Определение фильтра файлов для конкретного формата ISOBMFF

Спецификация фильтра для файлов в формате ISOBMFF определяется в таблице 9.6.2.4.5.2.2-1.

Таблица 9.6.2.4.5.2.2-1 – Спецификация фильтра файлов в формате ISOBMFF

```
#define MaxFilterFile 16    // максимальное количество байтов в блоке, которое фильтруется
#define MaxContainers  4    // максимальное количество блоков контейнера для блока
#define MaxUuidLen    16    // длина UUID в байтах

typedef struct BoxSpec {
    uint boxType                    // код 4CC типа блока
    byte extendedType[MaxUuidLen]
                                    // UUID для boxType=='uuid', в иных случаях значащие
символы отсутствуют
    byte filter[MaxFileFilter];     // соответствует байтам следующего блока
    byte filterMask[MaxFilter];
    ushort dataLen  ;               // максимальное количество данных блока, которое должно
быть получено
} 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;
}
```

Хост ECI должен проводить анализ файла и обнаруживать блоки, которые соответствуют полю **box** и содержатся в блоках, соответствующих любому из массивов **контейнера**. **Хост ECI** пропускает сканирование блоков, не определяемых в ISO/IEC 14496-12] или [ISO/IEC 23001-7].

Значение **boxType** в поле **container** в **dcrFileFilterIsobmff контейнера** может быть заменено подстановочными знаками '****'. В этом случае другие поля **контейнера** не должны быть значащими, а должны содержать нулевое значение, означающее отсутствие соответствия.

Поля **filter** и **filterMask** в **BoxSpec** должны быть сопоставлены с первыми байтами после поля type в обрабатываемом блоке. В полных блоках (см. [ISO/IEC 14496-12]) это поле version и flag. Сопоставление должно выполняться согласно функции boxMatch, при этом значение параметра **boxLen** задается равным количеству байтов, следующих за полями boxtype и extended_type в блоке, параметр **boxData** соответствует началу этих байтов, параметр **filter** равен значению поля **boxSpec.filter**, а параметр **filterMask** равен значению поля **boxSpec.filterMask**.

Возвращаемые фильтром данные представляют собой блоки (в последовательности), которые соответствуют фильтру при анализе файла **хостом ECI**. **Хост ECI** может объединять блоки в группы для удобства, однако при этом не должны возникать нежелательные задержки передачи блоков **клиенту ECI**, поскольку это может помешать **клиенту ECI** генерировать необходимые ключи дескремблирования.

9.6.2.4.5.2.3 Сообщение reqDcrFileFilter

C→H setDrcFileFilter(ushort **mH,** uchar **filterNr,** dcrFilleFilterSpec ***dataFilter)**

- Это сообщение отправляет **хосту ECI** запрос на установку фильтра данных на основе dataFilter с целью сбора данных по безопасности для **клиента ECI**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
filterNr: uchar	Номер фильтра файлов в **хосте ECI**
dataFilter: dcrFilleFilterSpec *	Спецификация фильтра для извлечения данных

Постусловие (запрос)

- Фильтр секций вводится в действие **хостом ECI** до выполнения команд **reqDcrFileStop** или **reqDcrFileQuit** либо до установки параметра reqDcrFileFilter в значение **dataFilter == NULL**.

9.6.2.4.5.2.4 Сообщение reqDcrFileAcqData

H→C reqDcrFileAcqData(ushort **mH,** uchar **filterNr,** uint **dataLen**, byte **data[]) →**
C→H resDcrFileAcqData (ushort **mH,** uchar **filterNr)**

- Это сообщение отправляет **хосту ECI** запрос на обнаружение и отправку **клиенту ECI** одной или нескольких секций в контексте медиафайла или потока, идентифицируемого указателем **mH**, и фильтра, идентифицируемого показателем **filterNr**.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** файла, для которого задается фильтр секций по умолчанию
filterNr: uchar	Номер программируемого фильтра. Значение находится в диапазоне от 0 до 7
dataLen: uint	Количество байтов в **данных**
data[]: byte	Последовательности секций private_sections (байтов в сетевом порядке) определяются в разделе 2.4.4.11 [ISO/IEC 13818-1-1]. Секция, содержащая ошибку CRC, не передается **клиенту ECI**

Определение параметра отклика

mH: ushort	**Указатель медиаданных** медиафайла или потока
filterNr: uchar	Номер запрограммированного фильтра

Соответствующие коды ошибок перечисляются в таблице 9.6.2.4.5.2.4-1.

Таблица 9.6.2.4.5.2.4-1 – Коды ошибок reqDerFileAcqData

Название	Описание
ErrDcrAcqDataTimeout	См. таблицу 9.6.2.4.7-1
ErrDcrAcqDataDataErr	

9.6.2.4.6 API контрольных слов дескремблирования файлов

9.6.2.4.6.1 Общие сведения

Секция API, выполняющая дескремблирование контента, позволяет **клиенту ECI** сделать доступным ключ для дескремблирования. **Хост ECI** должен в первую очередь обеспечить доступность контрольного слова, передавая Key-ID **клиенту ECI**. Как только ключ доступен, **хост ECI** может применять рассчитанное контрольное слово к (зашифрованному) контенту. Сообщения API, относящиеся к API дескремблирования контента файла **указателя медиаданных**, перечисляются в таблице 9.6.2.4.6.1-1.

Таблица 9.6.2.4.6.1-1 – API дескремблирования контента файла указателя медиаданных

Сообщение	Тип	Направление	Маркер	Описание
reqDcrFileKeyComp	A	H→C	0x20	Инициирование любого требуемого расчета или других действий **клиента ECI** для представления контрольного слова вместе с ключом Key-ID

9.6.2.4.6.2 Требования к обработке для хоста ECI

9.6.2.4.6.2.1 Контент в формате ISOBMFF CENC

В настоящем разделе определены требования к обработке для **хоста ECI**, выполняющего дескремблирование контента в формате ISOBMFF + CENC.

Хост ECI отвечает за своевременную передачу **клиенту ECI** любой информации по KeyID, с тем чтобы **клиент ECI** мог извлечь/обнаружить требуемое контрольное слово в надлежащее время. Другие ограничения, разрешающие эти операции, должны действовать как минимум за 30 секунд до предполагаемого использования контрольного слова.

Информация по Key-ID содержится в нескольких блоках, связанных с выборками медиаданных (последовательностями (частично) зашифрованных медиаданных): см., например, пункт 5.4 [b-DASH-IF V3]. Данные, содержащиеся в этих блоках, позволяют извлекать идентификаторы Key-ID, IV, а также позволяют идентифицировать открытые и зашифрованные данные в выборках медиаданных.

9.6.2.4.6.2.2 Контент в формате MPEG DASH

Подробная информация по форматам MPEG DASH, которые должен поддерживать **хост ECI**, в настоящее время не приводится в спецификациях **ECI**.

9.6.2.4.6.3 Сообщение reqDcrFileKeyComp

H→C reqDcrFileKeyComp(ushort **mh,** byte **keyId**[MaxUuidLen]**)** →
C→H resDcrFileKeyComp(ushort **mH)**

- Это сообщение инициирует вычисления и другие действия, необходимые **клиенту ECI** для расчета контрольного слова, идентифицируемого по KeyId, и предоставления доступа к нему для дешифрования контента.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** транспортного потока
keyId[MaxUuidLen]**:** byte	KeyID является идентификатором UUID в сетевом порядке байтов

Определение параметра отклика

mH: ushort	**Указатель медиаданных** транспортного потока

Предварительные условия (отклик)

1) Ключ доступен или произошла ошибка в период ожидания.

Подробная семантика

- **Клиент ECI** сообщает об ошибке в том случае, если запрашиваемое контрольное слово не может быть предоставлено своевременно (в течение 60 секунд). **Клиенты ECI** могут продолжать попытки получения запрашиваемого ключа и после сообщений об ошибке.
- После получения отчета об ошибке **хост ECI** может повторно создать **запрос**. **Хосты ECI** могут создавать не более 10 **запросов**.

Соответствующие коды ошибок перечисляются в таблице 9.6.2.4.6.3-1.

Таблица 9.6.2.4.6.3-1 – Коды ошибок reqDcrFileKeyComp

Название	Описание
ErrDcrFileUserDelay	См. таблицу 9.6.2.4.7-1
ErrDcrFileCardMissing	
ErrDcrFileServiceMissing	
ErrDcrFileResourceMissing	
ErrDcrFileMmiMissing	
ErrDcrFileKeyIdUnknown	
ErrDcrFileKeyOverflow	

9.6.2.4.7 Коды ошибок для API дешифрования контента на основе файлов и потоков

Значения конкретных ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечисляются в таблице 9.6.2.4.7-1.

Все запросы **указателей медиаданных,** связанных с файлами, возвращают код ошибки для параметра **указателя медиаданных** в том случае, если они применяются к **указателю медиаданных**, не связанному с файлами.

Таблица 9.6.2.4.7-1 – Коды ошибок API сеанса передачи медиаданных для файловых и потоковых медианосителей

Название	Значение	Описание
ErrDcrFileUserDelay	–256	Произошла длительная задержка при ожидании ввода данных **пользователем**, необходимых для завершения операции. Операция не завершена
ErrDcrFileCardMissing	–257	Смарт-карта, требуемая для сеанса, недоступна или не готова
ErrDcrFileServiceMissing	–258	Услуга, не относящаяся к оборудованию CPE (например, сервер DRM) и необходимая для выполнения клиентом ECI операций дешифрования, недоступна
ErrDcrFileResourceMissing	–259	Неизвестный ресурс в составе оборудования CPE, необходимый для доступа к контенту или для его дешифрования, недоступен
ErrDcrFileMmiMissing	–260	Доступ клиента ECI к интерфейсу MMI отсутствует
ErrDcrFileDescrContinue	–261	Хост ECI продолжает попытки дескремблирования контента в данном файле
ErrDcrAcqDataTimeout	–262	Произошла задержка при сборе данных
ErrDcrAcqDataDataErr	–263	В период задержки были получены секции, однако имеются ошибки. Как правило, это означает, что файл поврежден или не соответствует применимым спецификациям
ErrDcrFileKeyIdUnknown	–300	Идентификатор keyId не известен клиенту ECI / системе безопасности для данного контента
ErrDcrFileKeyOverflow	–301	Слишком много запросов Key-ID за короткий период; ожидание откликов клиента ECI на предыдущие запросы на обработку
ErrDcrFileKeyWithdrawn	–302	Ключ больше недоступен; права аннулированы клиентом ECI

9.7 Интерфейсы API для доступа к ресурсам повторного шифрования хоста ECI

9.7.1 Интерфейсы API повторного шифрования. Введение

9.7.1.1 Список интерфейсов API приводится в пункте 9.7

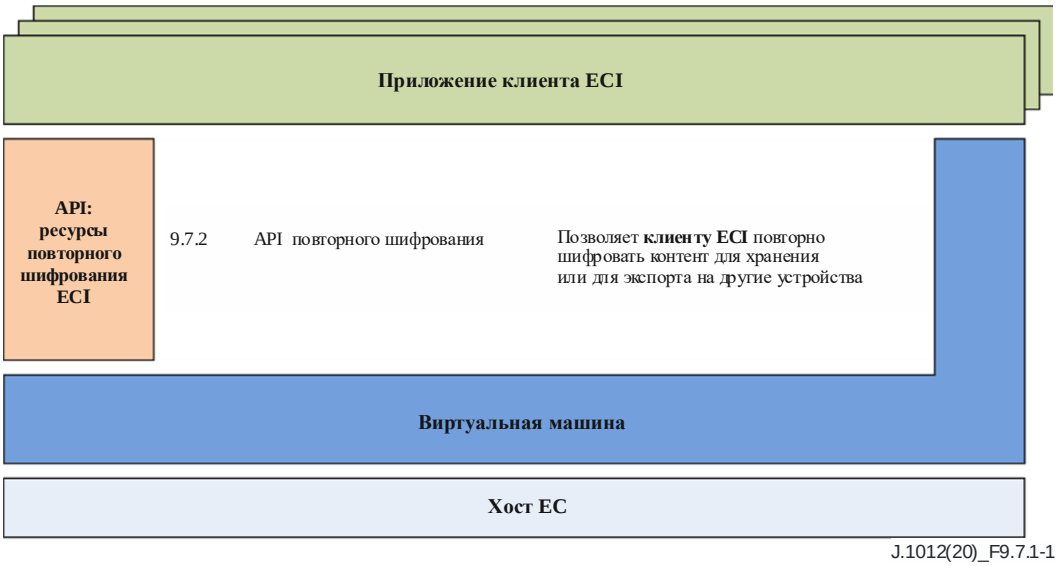


Рисунок 9.7.1-1 – Блок-схема интерфейсов API, определяемых в пункте 9.7

В таблице 9.7.1-1 перечисляются интерфейсы API, рассматриваемые в пункте 9.7, а рисунок 9.7.1-1 иллюстрирует расположение интерфейсов API, определяемых в пункте 9.7, а также **архитектуру ECI**. См. также [b-Menezes].

Таблица 9.7.1-1 – Список интерфейсов API, определяемых в пункте 9.7

Пункт	Наименование API	Описание
9.7.2.3	API **соединения экспорта**	Позволяет **клиенту ECI** устанавливать **соединение экспорта** для экспортируемого контента
9.7.2.5	API **соединения импорта**	Позволяет **клиенту ECI** импортировать контент, передаваемый в зашифрованном формате через сети доступа и дешифруемый под управлением **клиента ECI**
9.7.2.6	API дешифрования **микроклиента**	Позволяет **клиенту ECI** выполнять дешифрование импортируемого и повторно зашифрованного контента

9.7.1.2 Общие принципы повторного шифрования

Повторное шифрование в интерфейсе **ECI** позволяет независимой **микросистеме DRM** защищать контент, который передается **клиентом ECI** с помощью систем CA или DRM с целью последующего применения в оборудовании **CPE** или за его пределами. Система повторного шифрования в конфигурации, совместимой с **ECI**, называется **микросистемой DRM**. Примерами приложений **микросистемы DRM** могут служить системы со смещением по времени, PVR и потоковые передачи. **Клиент ECI**, выполняющий повторное шифрование контента, называется **микросервером**. **Микроклиентом** называется клиент, поддерживающий или не поддерживающий интерфейс **ECI**, способный дешифровать повторно зашифрованный контент. Образ клиента и идентификационные данные для повторного шифрования могут быть загружены в качестве типового **клиента ECI**, предоставленного главным микросервером DRM. На рисунке 9.7.1.2-1 изображен общий вид системы (за исключением главного микросервера DRM). При использовании локального хранилища **микросервер** и **микроклиент** реализуются как единое устройство.

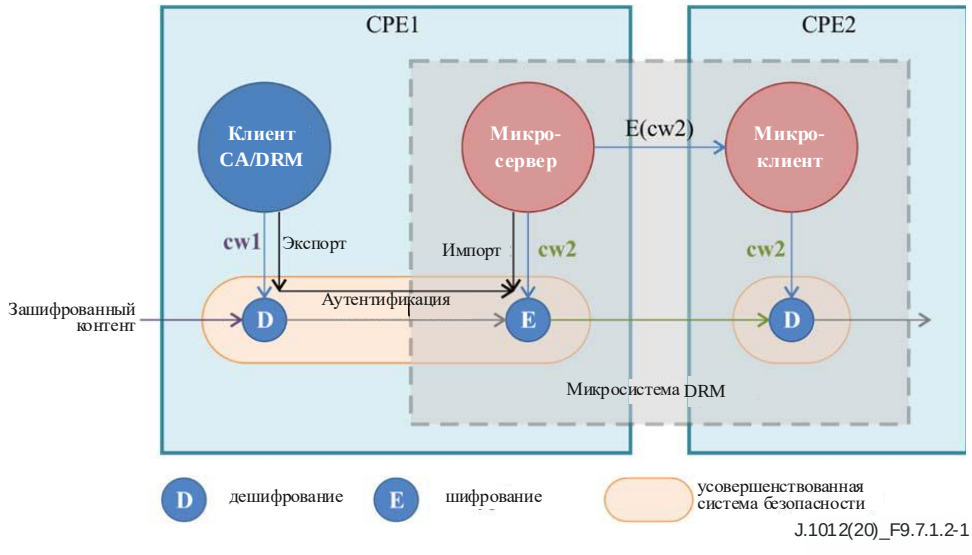


Рисунок 9.7.1.2-1 – Схема микросистемы DRM

Клиент ECI, использующий системы CA/DRM и на начальном этапе дешифрующий контент, может контролировать, разрешен ли экспорт контента в установленные **микросистемы DRM**. С этой целью **клиент ECI** выполняет аутентификацию **микросервера** посредством **усовершенствованной системы безопасности**; аутентификация проводится под контролем оператора CA/DRM. С момента окончания экспорта контента **микросистема DRM** несет ответственность за защиту контента. **Усовершенствованная система безопасности** обеспечивает безопасность дешифрования, повторного шифрования и аутентификации экспорта. Указанные принципы иллюстрируются на рисунке 9.7.1.2-1.

9.7.1.3 Обзор структуры API повторного шифрования

На рисунке 9.7.1.3-1 представлена более подробная схема, объясняющая роль различных интерфейсов API, задействованных в повторном шифровании. **Хост ECI** предоставляет **клиенту ECI**, выполняющему декодирование, всю необходимую информацию посредством API дешифрования. **Клиент ECI**, выполняющий декодирование, в защищенном режиме назначает контрольное слово для дешифрования контента посредством API усовершенствованной системы безопасности. Значимые особенности контента (метки) проходят аутентификацию. API экспорта разрешает **хосту ECI** направлять **клиенту ECI**, выполняющему декодирование, запрос на установление **соединения экспорта** с намеченным **микросервером** для выполнения повторного шифрования. API усовершенствованной системы безопасности позволяет **клиенту ECI**, выполняющему экспорт, проводить аутентификацию **микросервера**, выполняющего импорт. **Хост ECI** применяет API импорта для установления авторизованного **соединения экспорта** с **микросервером**. API повторного шифрования разрешает **хосту ECI** переводить микросервер в режим работы, соответствующий формату упаковки контента и приложению (потоковая передача, смещение во времени или хранение данных), и шифровать контент, предназначенный для выбранного (аутентифицированного) целевого **микроклиента**.

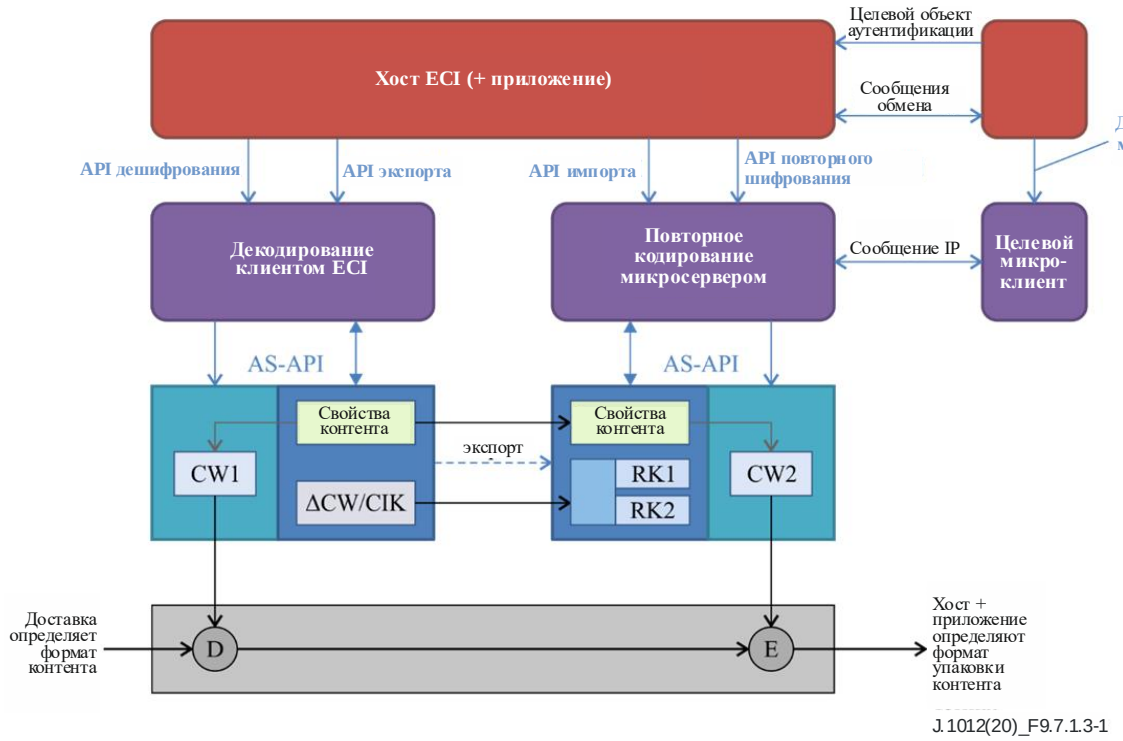


Рисунок 9.7.1.3-1 – Архитектура функциональных возможностей дешифрования и повторного шифрования

На рисунках 9.7.1.3-1 и 9.7.1.3-2 схематично представлены основные сообщения, применяемые в интерфейсах API дешифрования, управления экспортом, управления импортом, повторного шифрования и дешифрования микроклиента. На рисунке изображены слева направо этапы передачи контента: от **клиента ECI**, обеспечивающего доставку с помощью CA/DRM по **соединению экспорта/импорта** до **микросервера**, который шифрует дешифрованный контент, в заключение декодируемый **целевым микроклиентом**.

Четыре интерфейса API "хост–клиент" поддерживают следующие этапы обработки:

- На *этапе обнаружения* **клиенты ECI** могут объявлять потенциально возможные способы взаимодействия с **хостом ECI** (совместно с приложением). Это дает возможность **хосту ECI** обеспечивать соответствие запрашиваемого контента с конкретным **клиентом ECI**. Если выбранный **клиент ECI** не обладает соответствующими правами на обработку данного контента, **хост ECI** должен найти других **клиентов ECI**. В домашних сетях и распределенных приложениях PVR для этого предусмотрены протоколы приложений, такие как DLNA, см. [b-DLNA]. *Этап аутентификации* позволяет **хосту ECI** устанавливать аутентифицированное соединение между требуемым **клиентом ECI** и **микросервером** или **микросервером** и **микроклиентом**. Аутентификация может быть неявной, то есть криптографическая проверка, выполняемая для аутентификации, заключается в способности **клиента ECI** дешифровать контент на заключительном этапе. За аутентификацией всегда следует поток контента. В некоторых случаях требуется обратное соглашение. Для решения коммерческих задач **соединению импорта** может потребоваться одобрение **микросервера**.
- На *этапе создания сеанса* **хосту ECI** разрешается резервировать все ресурсы, необходимые для дешифрования или шифрования контента в определенном режиме работы, связанном с **указателем медиаданных**. Соединения импорта и **целевые соединения** для reqEncrMhOpen определяются на **микросервере**, или предполагаются в **клиенте ECI** CA/DRM. Следует отметить, что **хост ECI** отвечает за распределение дополнительных ресурсов, таких как ресурсы (де)скремблирования, демультиплексирования и декодирования, что необходимо для осуществления общего сценария медиаприложений. **Клиент ECI** в конечном счете направляет запрос на присвоение ресурсов AS и дешифрования или шифрования, используя API **усовершенствованной системы безопасности**.

- На *этапе управления сеансом* **хосту ECI** разрешается запускать и останавливать обработку контента на основе **указателей медиаданных**. Для непрерывной обработки контента на трассе передачи требуется запускать **клиентов ECI** от места назначения до источника; то есть **клиент ECI** должен быть готов к обработке контента в том виде, в котором он представлен.

Фаза протокола	**Клиент передачи CA/DRM**		**Микро-сервер**		**Микроклиент**
	Хост-> Клиент	Клиент <-Хост	Хост-> Клиент	Клиент <-Хост	Хост-> Клиент
Интерфейс API	***Дешифрование***	***Контроль экспорта***	***Контроль импорта***	***Повторное шифрование***	***иС дешифрование***
Обнаружение	setDcrMhMatch	reqExpConnNodes	reqImpConnNodes reqImpConnChain	reqEncrTargets	reqDcrTargets reqDcrTargetCred
Аутентификация	(процедура предоставления услуг	reqExpConnSetup reqExpConnDrop *reqExpConnCancel*	reqImpConnSetup reqImpConnDrop *reqImpConnCancel*	reqEncrConnSetup reqEncrConnDrop reqEncrConnCancel	
Создание сеанса	reqDcrMhOpen reqDcrMhClose reqDcrMhCancel	reqExpMhOpen reqExpMhClose reqExpMhCancel	(выполняется при помощи сообщения повторного шифрования)	reqEncrMhOpen reqEncrMhClose reqEncrMhCancel	reqDcrMhOpen reqDcrMhClose reqDcrMhCancel
Управление сеансом	reqDcrTsStart reqDcrTsStop reqDcrTsQuit reqDcrFileStart reqDcrFileStop reqDcrFileQuit			reqEncrMhStart reqEncrMhStop reqEncrMhQuit	reqDcrTsStart reqDcrTsStop reqDcrTsQuit reqDcrFileStart reqDcrFileStop reqDcrFileQuit

J.1012(20)_F9.7.1.3-2

Рисунок 9.7.1.3-2 – Общая схема API шифрования/дешифрования и импорта/экспорта

Сообщения используют определенный порядок семантики и присвоения имен:

- На *этапе обнаружения* **клиенту ECI** разрешается заявлять свои возможности для подключения к другому **клиенту ECI** или контенту. Сообщения setDcrMhMatch, reqExpConnNodes, reqImpConnNodes, reqEncrTargets и reqDcrTargets направляют **клиенту ECI** запрос на объявление этих возможностей (в форме идентификаторов).
- На этапе аутентификации применяются сообщения *setup*, *drop* и *cancel* для создания (аутентифицированного) соединения, перераспределения предыдущего соединения или отмены подобного соединения **клиентом ECI**. Опорным элементом для соединения являются **соединение экспорта** (**клиент ECI**, выполняющий экспорт контента), **соединение импорта** (**клиент ECI**, выполняющий импорт контента) или **целевое** соединение (**микросервер**, шифрующий контент для последующего дешифрования, выполняемого **целевым объектом** и наоборот, например, **микроклиент**, дешифрующий контент от **микросервера**).
- Для создания и завершения всех сеансов, ссылающихся на **указатель медиаданных** как на общую точку отсчета, на *этапе создания сеанса* применяются сообщения *open*, *close* и *cancel*. Кроме того, требуемое **клиентом ECI** управление сеансами MMI и ресурсами **смарт-карт** может ссылаться на **указатель медиаданных**, чтобы разрешать **хосту ECI** привязывать запрос диалога **пользователя** в рамках приложения.
- На *этапе управления сеансом* определяются различные сообщения для дешифрования двух конкретных форматов контента: транспортных потоков и файлов. Обработка может быть *запущена* или остановлена **хостом ECI**; а в случае недостатка ресурсов или проблем с правами **клиент ECI** может *прекратить* обработку.

ПРИМЕЧАНИЕ 1. – Возможно, что некоторые системы защиты не должны в обязательном порядке выполнять все этапы обработки. **Клиенты ECI** таких систем могут выполнять только незначительную административную обработку некоторых сообщений.

ПРИМЕЧАНИЕ 2. – Свойства **клиентов ECI** в **соединениях импорта/экспорта** отличаются от характера взаимодействия между **микросервером** и **микроклиентом**. **Клиенты ECI** в **соединениях импорта/экспорта** совместно используют **хост ECI** и способны обмениваться контентом на основе механизма экспорта AS, используя определенные **цепочки сертификатов** импорта/экспорта **ECI**. **Микросервер** и **микроклиент** могут применять протокол выбора (характерный для **микросистемы DRM**) для установления соединений при условии соответствия структуре API, а также использовать **систему AS** для проведения аутентификации и формирования общих ключей. Обмен контентом в **соединениях экспорта/импорта** носит неявный характер (определяемый **хостом ECI**); подлинность **микросервера** (необходимая для целей экспорта), подтверждается **системой AS**. Для обмена контентом между **микросервером** и **микроклиентом** требуется сеанс указателя **медиаданных** и управление сеансом как на **микросервере**, так и на **микроклиенте**.

9.7.2 API управления экспортом ECI

9.7.2.1 Введение

ECI позволяет **клиентам ECI** экспортировать декодированный контент на **микросервер**, который обеспечивает повторное шифрование для целей (разрешенного) перераспределения на другие устройства или (разрешенного) хранения контента для последующего воспроизведения. С этой целью интерфейс **ECI** задает структуру **сертификата**, которая определяет группы **микросистем DRM**, в которые разрешен экспорт. Каждый декодированный элемент контента сопровождается идентификацией соответствующей **группы экспорта**. От **группы экспорта** должна существовать цепочка **сертификатов**, разрешающих экспорт на выбранный **микросервер**. Эта цепочка обрабатывается усовершенствованной системой безопасности, что позволяет предоставить весьма устойчивый механизм авторизации экспорта.

Клиент ECI, выполняющий экспорт, отвечает за предоставление **сертификатов группы экспорта** и всех прямых преемников. **Микроклиент**, выполняющий импорт, отвечает за предоставление дополнительной информации по учетным данным, которая позволяет завершить цепочку от экспортирующего **клиента ECI** до импортирующего.

Хост ECI может устанавливать соединение для повторного шифрования от **клиента ECI**, выполняющего дешифрование, до **микросервера**, выполняющего шифрование. После установления соединения **хост ECI** может продолжать дешифрование и повторное шифрование контента с использованием сеансов **указателя медиаданных**. **Система AS** обеспечивает защищенную передачу контента и связанной с ним информации по безопасности от **клиента ECI**, выполняющего декодирование, **микроклиенту** на основе предоставленных через **систему AS** учетных данных.

Хосты ECI предоставляют **клиентам ECI** поддержку по обеспечению доступа к сетевым услугам с целью получения актуальных идентификационных данных для экспорта и импорта, например, через API карусели передачи данных (пункт 9.5.4) и интерфейс API HTTP IP (пункт 9.4.4.6).

В целях повторного шифрования **хост ECI** и приложение должны устанавливать авторизованных **микроклиентов**, которым разрешается декодировать контент. Это может быть как отдельное оборудование **CPE** (с соответствующим клиентом), так и группа (на основе совместно используемого ключа). Затем **хост ECI** устанавливает авторизованное соединение между **микросервером** и соответствующим ему **микроклиентом** (одно соединение для каждого **микроклиента**). В приложениях с временным сдвигом и записью информация, необходимая **клиенту ECI** для выполнения в дальнейшем декодирования контента, может быть сохранена (например, вместе с повторно зашифрованным контентом). В соединениях с потоковой передачей в режиме реального времени сообщения управления сеансом необходимые **микросерверу** и **микроклиенту**, могут передаваться либо через **хост ECI**, если **микроклиенты** и **микросервер** располагаются на одном устройстве, либо непосредственно между **микроклиентами** через IP-соединение.

ПРИМЕЧАНИЕ. – Протоколы связи и соответствующие аспекты безопасности между **клиентами ECI** выходят за рамки сферы применения **ECI**.

9.7.2.2 Структуры сертификатов экспорта

9.7.2.2.1 Общая структура

Механизм экспорта **ECI** основывается на **сертификатах**. Большинство **сертификатов** имеют соответствующий **список аннулирования**, допускающий обновления разрешений на экспорт. На рисунке 9.7.2.2.1-1 представлена структура **сертификата** для непосредственного контроля за экспортом **клиента ECI**, выполняющего декодирование.

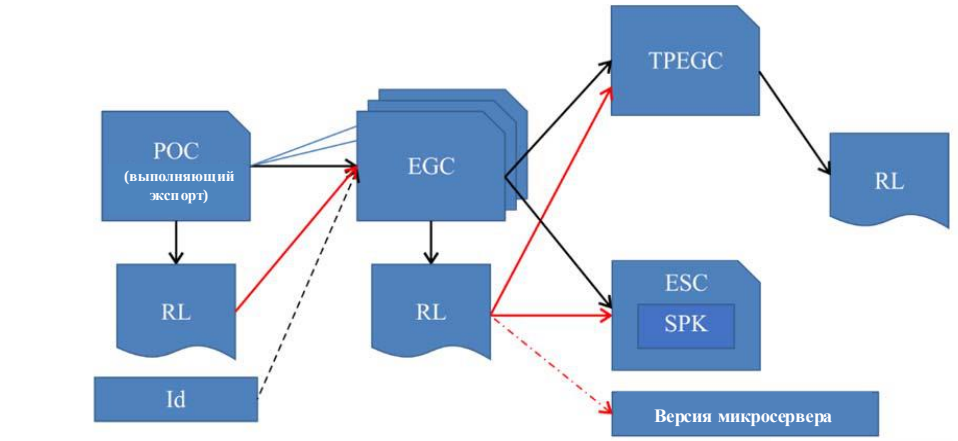


Рисунок 9.7.2.2.1-1 – Структура распределения сертификатов ECI

Сертификат **управления платформой клиента ECI** (POC) является **родительским объектом** сертификатов **группы экспорта**. POC **ECI** имеет специальный список аннулирования, позволяющий **клиенту ECI** управлять сертификатом **группы экспорта** и связанными с ним версиями списка аннулирования. Каждый сертификат **группы экспорта** является **родительским объектом** действующих сертификатов экспорта или последующей **группы экспорта** (преемника). Существуют два типа сертификатов экспорта:

1) Сертификат системы экспорта (ESC) с помощью собственного **открытого ключа отправителя** идентифицирует **микросервер**, на который разрешен экспорт, что позволяет выполнить немедленную аутентификацию. Кроме того, номер версии списка аннулирования ESC используется для определения номера минимальной версии **микросервера**.

2) Сертификат **группы экспорта** третьей стороны (TPEGC) относится к **сертификату группы экспорта**, которым управляет другая организация. Это позволяет аутентифицировать более крупные неоднородные группы **микросистем DRM** с помощью одного **сертификата** экспорта.

Структура сертификата группы экспорта третьей стороны подробно показана на рисунке 9.7.2.2.1-2.

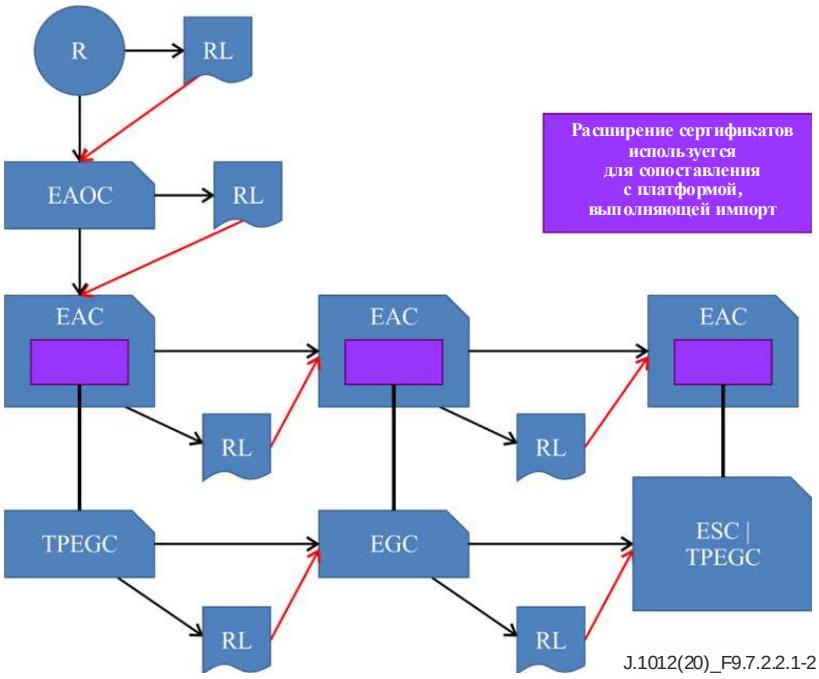


Рисунок 9.7.2.2.1-2 – Структура сертификата группы экспорта третьей стороны

Корневой сертификат ECI является **родительским объектом** операторского сертификата авторизации экспорта (EAOC). **Корневой сертификат ECI** содержит специальный список аннулирования для таких **сертификатов**. Операторский сертификат авторизации экспорта (EAOC) является **родительским объектом** сертификата авторизации экспорта (EAC). Этот **сертификат** соответствует сертификату **группы экспорта** третьей стороны (TPEGC). Данный механизм позволяет осуществлять двойную аутентификацию группы третьей стороны для обеспечения дополнительной безопасности.

Сертификат **группы экспорта** третьей стороны является **родительским объектом** для:

1) **сертификата группы экспорта** (EGC), который сам может являться **родительским объектом другого** EGC, или любого из перечисленных ниже **сертификатов**. Каждый EGC имеет соответствующий **список аннулирования**;
2) сертификата **системы** экспорта (ESC);
3) (следующего) сертификата **группы экспорта** третьей стороны (TPEGC).

Каждый **сертификат** дополнительно проверяется с использованием соответствующего сертификата авторизации экспорта (EAC); тем самым формируется дерево, соответствующее дереву TPEGC/EGC.

В таблице 9.7.2.2.1-1 приводится обзор **сертификатов** и их **родительских объектов**.

Таблица 9.7.2.2.1-1 – Краткие сведения о различных сертификатах экспорта

Название сертификата	Сокращение	Описание	Родительский объект
Группа экспорта	EGC	Этот **сертификат** позволяет **клиентам ECI**, выполняющим экспорт, идентифицировать набор (группу) **микроклиентов** и/или аутентифицированные группы третьей стороны, которым они разрешают экспорт. Применимая **группа экспорта** определяется как часть аутентифицированного атрибута прав для контента	POC, TPEGC, EGC
Группа экспорта третьей стороны	TPEGC	**Сертификат** для аутентификации **группы микросистем DRM**, управляемый другой (третьей) стороной	EGC, TPEGC
Оператор авторизации экспорта	EAOC	**Сертификат**, предоставляющий основу для оператора, который оказывает услуги авторизации для **групп экспорта** третьей стороны. Данный сертификат является **родительским объектом** деревьев сертификата авторизации экспорта для **групп экспорта** третьей стороны, для которых он выполняет совместную аутентификацию	**Корневой элемент ECI**
Авторизация экспорта	EAC	Этот **сертификат** обеспечивает совместную аутентификацию сертификата **группы экспорта** третьей стороны или сертификата **группы экспорта**, который управляется третьей стороной	EAC, EAOC
Система экспорта	ESC	Этот **сертификат** аутентифицирует сертификат **системы управления платформой микроклиента**	EGC, TPEGC

9.7.2.2.2 Определения сертификатов экспорта

9.7.2.2.2.1 Сертификат группы экспорта и список аннулирования

Определения **сертификатов групп экспорта ECI** (EGC) должны соответствовать общему определению формата ECI_Certificate, приведенному в пункте 5.2. EGC использует поле идентификатора **сертификатов ECI** со следующим определением поля, данным в таблице 9.7.2.2.2.1-1.

Таблица 9.7.2.2.2.1-1 – Определение идентификатора группы экспорта ECI

Синтаксис	Количество битов	Мнемоника
`ECI_EGC_Id {`		
`    type   /* см. таблицу 5.3-1*/`	4	uimsbf
`    export_group_id /* см. таблицу 5.3-1 */`	20	uimsbf
`    export_group_version`	8	uimsbf
`}`		

Семантика

Type	Значение в соответствии с таблицей 5.2-1
export_group_id: integer	Идентификатор, присвоенный **группе экспорта** объектом, управляющим **группой экспорта**. Значения 0x00000 и 0xFFFFF0-0xFFFFF резервируются
export_group_version: integer	Версия **сертификата группы экспорта** с идентификатором **export_group_id**

В целях аутентификации **дочерних сертификатов** EGC должен сопровождаться списком аннулирования в соответствии с пунктом 5.3, в частности таблицей 5.3-1.

9.7.2.2.2.2 Сертификат группы экспорта третьей стороны и список аннулирования

Определения **сертификатов для групп экспорта** третьей стороны **ECI** (TPEGC) должны соответствовать общему определению поля ECI_Certificate, приведенному в пункте 5.2. TPEGC использует поле идентификатора **сертификатов ECI**. Определение поля приводится в таблице 9.7.2.2.2.2-1.

Таблица 9.7.2.2.2.2-1 – Определение поля идентификатора TPEGC

Синтаксис	Количество битов	Мнемоника
ECI_TPEGC_Id {		
type /* см. таблицу 5.2-1***/**	4	uimsbf
tp_export_group_id /* см. таблицу 5.3-1 ***/**	20	uimsbf
tp_export_group_version	8	uimsbf
}		

Семантика

Type	Значение в соответствии с таблицей 5.3-1
tp_export_group_id: integer	Идентификатор, присвоенный **группе экспорта** третьей стороны объектом, управляющим **группой экспорта** третьей стороны. Значения 0x00000 и 0xFFFFF0-0xFFFFF резервируются
tp_export_group_version: integer	Версия стороннего **сертификата группы экспорта** с идентификатором **tp_export_group_id**

Поле расширения TPEGC, как указывается в таблице 9.7.2.2.2.2-2, должно содержать следующую структуру с использованием определений **export_authorization_operator_id** в таблице 9.7.2.2.2.4-1 и **export_authorization_id** в таблице 9.7.2.2.2.5-1.

Таблица 9.7.2.2.2.2-2 – Определение поля расширения

Синтаксис	Количество битов	Мнемоника
ECI_ TPEGC _Extension {		
export_authorization_operator_id	20	uimsbf
export_authorization_id	20	uimsbf
padding(4)		
Extension_field **extension**		
}		

Семантика

export_authorization_operator_id: integer	Идентификатор **ECI сертификата оператора а**вторизации экспорта, который совместно выполняет аутентификацию этого **сертификата**
export_authorization_id: integer	Идентификатор **ECI сертификата** авторизации экспорта, который совместно выполняет аутентификацию этого **сертификата** (см. пункт 9.7.1.2.2.5)
extension: Extension_field	Расширение данной структуры

В целях аутентификации **дочерних сертификатов** TPEGC должен сопровождаться списком аннулирования в соответствии с пунктом 5.3 и таблицей 5.3-1.

9.7.2.2.2.3 Корневой список аннулирования для сертификатов оператора авторизации экспорта

В целях аутентификации цепочка аутентификации экспорта должна начинаться с корневого списка аннулирования в соответствии с пунктом 5.3 и таблицей 5.3-1.

9.7.2.2.2.4 Операторский сертификат авторизации экспорта

Определения **сертификатов оператора** авторизации экспорта **ECI** (EAOC) должны соответствовать общему определению поля ECI_Certificate, приведенному в пункте 5.2. EAOC использует поле идентификатора **сертификатов ECI**, определение которого дается в таблице 9.7.2.2.2.4-1.

Таблица 9.7.2.2.2.4-1 – Определение поля идентификатора EAOC

Синтаксис	Количество битов	Мнемоника
ECI_EAOC_Id {		
type **/* см.** таблицу 5.3-1***/**	4	uimsbf
export_authorization_operator_id **/* см.** таблицу 5.3-1 ***/**	20	uimsbf
export_authorization_operator_version	8	uimsbf
}		

Семантика

type	Значение в соответствии с таблицей 5.3-1
export_authorization_operator_id: integer	Идентификатор, присвоенный оператору авторизации экспорта Значения 0x00000 и 0xFFFFF0-0xFFFFF резервируются
export_authorization_operator_version: integer	Версия сертификата оператора авторизации экспорта с идентификатором **export_authorization_operator_id**

В целях аутентификации **дочерних сертификатов** EAOC должен сопровождаться списком аннулирования в соответствии с пунктом 5.3 и таблицей 5.3-1.

9.7.2.2.2.5 Сертификат авторизации экспорта и список аннулирования

Определения **сертификатов** авторизации экспорта **ECI** (EAC) должны соответствовать общему определению поля ECI_Certificate, приведенному в пункте 5.2, с использованием специального непустого поля расширения. EAC использует поле идентификатора **сертификатов ECI**. Определение поля дается в таблице 9.7.2.2.2.5-1.

Таблица 9.7.2.2.2.5-1 – Определение поля расширения EAC

Синтаксис	Количество битов	Мнемоника
ECI_EAC_Id {		
type **/* см.** таблицу 5.3-1***/**	4	uimsbf
export_authorization_id **/* см.** таблицу 5.3-1 ***/**	20	uimsbf
export_authorization_version	8	uimsbf
}		

Семантика

Type	Значение в соответствии с таблицей 5.3-1
export_authorization_id: integer	Идентификатор, присвоенный **сертификату** авторизации экспорта (в контексте его **родительского объекта**). Значения 0x00000 и 0xFFFFF0-0xFFFFF резервируются
export_authorization_version: integer	Версия сертификата авторизации экспорта с идентификатором **export_authorization_id**

Поле расширения EAC содержит структуру **сертификата**, которая должна быть авторизована для экспорта (см. пункт 5.1.3), за исключением поля **signature** (подпись), за которым следует поле расширения.

В целях аутентификации **дочерних сертификатов** EAC должен сопровождаться списком аннулирования в соответствии с пунктом 5.3 и таблицей 5.3-1, если это необходимо для аутентификации **дочерних** сертификатов.

9.7.2.2.2.6 Сертификат системы экспорта

Определения **сертификатов** системы экспорта **ECI** (ESC) должны соответствовать общему определению поля ECI_Certificate, приведенному в пункте 5.2. Поле public_key **сертификата** должно содержать значение SPK, используемое **микросервером**. ESC использует поле идентификатора **сертификатов ECI**. Определение поля дается в таблице 9.7.2.2.2.6-1.

Таблица 9.7.2.2.2.6-1 – Определение поля расширения ESC

Синтаксис	Количество битов	Мнемоника
ECI_ESC_Id {		
type **/* см.** таблицу 5.3-1**/**	4	uimsbf
export_system_id **/* см.** таблицу 5.3-1 ***/**	20	uimsbf
export_system_version	8	uimsbf
}		

Семантика

Type	Значение в соответствии с таблицей 5.3-1
export_system_id: integer	Идентификатор, присвоенный **сертификату** системы экспорта (в контексте его **родительского объекта**). Значения 0x00000 и 0xFFFFF0-0xFFFFF резервируются
export_system_version: integer	Версия **сертификата** системы экспорта с идентификатором **export_system_id**

9.7.2.2.3 Проверка цепочек сертификатов экспорта

Клиент ECI, выполняющий экспорт, с предварительно проверенной цепочкой и с дополнительными цепочками авторизации экспорта создает запрашиваемое **соединение импорта/экспорта**. **Клиент ECI**, выполняющий экспорт, и **микросервер ECI**, выполняющий импорт, ответственные за свои части цепочек, предоставляют **пользователю** информацию в случае возникновения проблем и/или попыток обнаружения обновленных цепочек. **Клиент ECI** предоставляет эти цепочки для обработки **системе AS**, чтобы установить необходимое **соединение экспорта/импорта**. Если **система AS** обнаруживает ошибки проверок в любой цепочке или в дополнительной авторизации экспорта, **клиент ECI** не может устанавливать требуемое соединение.

Сертификаты авторизации экспорта используются для совместной аутентификации **сертификата** экспорта. Правила обработки для совместной аутентификации:

1) Сертификат авторизации экспорта и **сертификат**, подлежащий совместной аутентификации, имеют действительные подписи (как определяется их соответствующими **родительскими объектами**) и не аннулируются.

2) Все данные в **сертификате**, подлежащем совместной аутентификации, за исключением его подписи, сравниваются с данными в соответствующем поле расширения **сертификата** авторизации экспорта. В случае несовпадения совместную аутентификацию выполнить не удается.

Для настройки **соединения экспорта** подсистема CPS должна следовать нижеприведенным правилам обработки:

1) Должны соблюдаться все правила обработки CPS для **цепочек сертификатов**, перечисляемых в пункте 5.4.2.

2) CPS проверяет типы д**очернего объекта родительского сертификата** на соответствие таблице 5.2-2.

3) Родительским **объектом** для **цепочки экспорта клиента ECI** должен являться POC **ECI** клиента. Сопутствующий список аннулирования для **групп экспорта** применяется для проверки **дочерних сертификатов группы экспорта**. Номер версии списка аннулирования POC для **групп экспорта** должен быть выше, чем для minClientVersion клиента (см. [ITU-T J.1014]).

4) Подсистема CPS должна принимать максимум 2 уровня сертификатов EGC для **клиента ECI**, выполняющего экспорт. То есть **дочерним сертификатом** второго уровня EGC должен являться сертификат TPEGC или ESC.

5) Подсистема CPS гарантирует, что любой TPGC сопровождается сертификатом EAC, прошедшим совместную аутентификацию через цепочку (с сопутствующими списками аннулирования) от корневого сертификата до EAOC и EAC. Версия корневого списка аннулирования для сертификата оператора авторизации экспорта используется в целях определения максимального номера версии списка аннулирования при "проверке целостности системы".

6) Подсистема CPS гарантирует, что любой EGC, ESC и TPEGC (в порядке убывания от TPEGC) совместно аутентифицируется EAC, который является **дочерним объектом** EAC, проверяющим **родительский объект** этого **сертификата**.

Клиенты **ECI** и **микросерверы DRM**, выполняющие экспорт, должны обеспечивать надлежащую предварительную обработку в своих цепочках и предоставлять последние доступные версии, чтобы избежать аннулирования в CPS.

9.7.2.2.4 Транспортные протоколы для идентификационных

9.7.2.2.4.1 Общие сведения

Клиенты ECI и **микросерверы**, выполняющие экспорт, могут определять собственные форматы для транспортировки идентификационных данных. Интерфейс **ECI** определяет стандартизованный формат файлов для передачи и подобных данных. **Клиентам ECI** могут получить доступ к этим стандартизированным файлам через API доступа карусели **ECI** для каналов радиовещания. Чтобы обеспечить онлайн-доступ для клиентов, в интерфейсе **ECI** определяются стандартные процедуры API-вызовов.

9.7.2.2.4.2 Формат файла дерева экспорта

Формат файлов для дерева **групп экспорта** определяется в таблице 9.7.2.2.4.2-1.

Таблица 9.7.2.2.4.2-1 – Определение файла дерева экспорта ECI

Синтаксис	Количество битов	Мнемоника
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**		
}		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – три 8-битовых представления символов 'EET' в формате ASCII. Клиенты ECI проверяют значение данного поля, чтобы удостовериться, что формат файла ECI соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; остальные значения версий резервируются. **Клиенты ECI** игнорируют все образы, номер версии которых не распознается
operator_id: ECI_Operator_Id	Идентификатор **оператора клиента ECI** дерева экспорта, содержащегося в файле Поле operator_version соответствует корневому элементу дерева export_group_tree
Platform-operation_id: ECI_Platform_Operation_Id	Идентификатор **системы управления платформой клиента ECI** дерева экспорта, содержащегося в файле
export_group_tree: ECI_RL_Tree	Структура ECI_RL_Tree начинается со списка аннулирования **для групп экспорта**. Если сертификаты не требуют наличия дополнительного **списка аннулирования**, данная структура содержит пустой **список аннулирования** с подписью, которая не требует соответствия **сертификату**
extensions: Extension_field	Дополнительные данные, определяемые оператором

9.7.2.2.4.3 Формат файлов цепочек импорта

Формат файлов для **цепочек импорта микросервера** определяется в таблице 9.7.2.2.4.3-1.

Таблица 9.7.2.2.4.3-1 – Определение файла цепочки импорта ECI

Синтаксис	Количество битов	Мнемоника
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**		
}		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – три 8-битовых представления символов 'EIC' в формате ASCII. **Клиенты ECI** проверяют значение данного поля, чтобы удостовериться, что формат файла **ECI** соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; остальные значения версий резервируются. **Клиенты ECI** игнорируют все образы, номер версии которых не распознается
operator_id: ECI_Operator_Id	Идентификатор **оператора микросервера**, для которого предназначена данная **цепочка импорта**
platform_operation_id: ECI_Platform_Operation_Id	Идентификатор **системы управления платформой микросервера**, для которого предназначена данная **цепочка импорта**
nr_chains: integer	Номер **цепочки импорта** в файле
eaoc_id: ECI_Operator_Id	Идентификатор **оператора** авторизации **цепочки импорта**
eac_id: ECI_Platform_Id	Идентификатор сертификата EAC, выполняющего совместную авторизацию системы управления платформой цепочки импорта
import_chain: ECI_Certificate_Chain	**Цепочка сертификатов ECI** от **сертификата систем управления платформой** импорта до ESG, идентифицирующей **микроклиента**. Цепочка может содержать несколько сертификатов TPEGC. Каждая действующая **цепочка импорта** должна быть представлена отдельно: то есть, если цепочка 1 состоит из двух субцепочек третьих лиц, а вторая субцепочка может также использоваться отдельно как **цепочка импорта**, она должна быть представлена отдельно. Если **сертификаты** не требуют наличия дополнительного **списка аннулирования**, данная структура содержит пустой **список аннулирования** с подписью, которая не требует соответствия **сертификату**
extensions: Extension_field	Дополнительные данные, определяемые оператором

9.7.2.2.4.4 Формат файлов авторизации экспорта

Формат файлов для авторизации **цепочек экспорта микросервера** определяется в таблице 9.7.2.2.4.4-1.

Таблица 9.7.2.2.4.4-1 – Определение файла авторизации экспорта ECI

Синтаксис	Количество битов	Мнемоника
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**		
}		
}		

Семантика

magic: byte[3]	Системный код, используемый для проверки формата следующих данных. Его значение – три 8-битовых представления символов 'EEA' в формате ASCII.**Клиенты ECI** проверяют значение данного поля, чтобы удостовериться, что формат файла **ECI** соответствует ожидаемому. Это позволяет обеспечить дополнительную целостность данных
image_header_version: byte	Версия формата заголовка образа. Значение 0x01 является текущей заданной версией; остальные значения версий резервируются. **Клиенты ECI** игнорируют все образы, номер версии которых не распознается
operator_id: ECI_Operator_Id	Идентификатор **оператора микросервера**, для которого предназначена данная **цепочка импорта**
Platform_operation_id: ECI_Platform_Operation_Id	Идентификатор **системы управления платформой микросервера**, для которого предназначена данная **цепочка импорта**
nr_chains: integer	Количество цепочек авторизации экспорта в данном файле
direct_flag: bit	Если значение равно 0b1, следующая цепочка выполняет непосредственную авторизацию субцепочки ESC, а идентификаторы **o_id** и **po_id** не имеют значения. Если значение равно 0b0, следующая цепочка выполняет авторизацию субцепочки TPEGC, а **o_id** и **po_id** представляют идентификаторы **сертификата** авторизации
o_id: ECI_Operator_Id	Идентификатор **оператора** третьей стороны в промежуточной **цепочке экспорта** третьей стороны, которая аутентифицируется следующей цепочкой аутентификации экспорта
po_id: ECI_Platform_Öperation_Id	Идентификатор **системы управления платформой** третьей стороны в промежуточной **цепочке экспорта** третьей стороны, которая аутентифицируется следующей цепочкой аутентификации экспорта
chain: ECI_Certificate_Chain	**Цепочка сертификатов ECI** от **корневого сертификата ECI** до сертификата EAC, аутентифицирующего первый TPEGC, ESG
extensions: Extension_field	Дополнительные данные, определяемые оператором

9.7.2.2.4.5 Радиовещательные карусели, передающие идентификационные данные экспорта

Операторы могут развертывать карусели, определяемые поддержкой **ECI**, как указывается в пункте 7.7.2, для передачи идентификационных данных **клиентов ECI** по экспорту и/или импорту, которые могут быть выбраны для поддержки. Однако для любого конкретного **клиента ECI хост ECI** должен лишь контролировать обновления отдельного DSI расположения карусели данных. То есть в целях передачи идентификационных данных экспорта или импорта с использованием стандартного формата карусели **оператор** использует для такого **клиента ECI** ту же карусель, что передает образ клиента, идентификационные данные **системы управления платформой**, данные аннулирования и т. д. См. также пункт 7.7.2.1.

Форматы данных модулей карусели должны соответствовать таблице 7.7.2.6-1. Модули, присвоенные дескриптором compatibilityDescriptor, с полем descriptorType, равным 0xB0, содержат модули с единственной структурой ECI_Export_Tree_File. Модули с полем descriptorType, равным 0xB1, содержат модули с единственной структурой ECI_Import_Chain_File. Модули с полем descriptorType, равным 0xB2, содержат модули с единственной структурой ECI_Export_Authentication_File.

Рекомендуется, чтобы контроль обновлений в карусели **клиентом ECI** совпадал с контролем, выполняемым **хостом ECI** для данных другого **клиента ECI**, что позволяет осуществлять эффективное управление электропитанием.

9.7.2.2.4.6 Предоставление идентификационных данных экспорта в режиме онлайн

В настоящей Рекомендации резервируются следующие структуры URL веб-интерфейса API, позволяющие стандартной структуре предоставлять **клиентам ECI** доступ к идентификационным данным экспорта из онлайн-сервера оператора.

Со ссылкой на пункт 7.7.3 указывается определение tail_extension и условных обозначений:

```
tail_extension* ::=
        client_export |
        client_import |
        client_exp_auth .
```

Условное обозначение tail_extension* указывает на то, что другие расширения могут присутствовать в будущих версиях настоящей Рекомендации.

Следующие запросы веб-интерфейса API определяются для импорта/экспорта:

```
client_export ::= 'client-export/' operator_id '/' platform_operation_id.
```

Этот запрос возвращает последнюю версию файла дерева экспорта в формате ECI_Export_Tree_File для **клиента ECI**, обозначенного идентификаторами **operator_id**, **platform_operatio_id**.

```
client_import ::= 'client-import/' operator_id '/' platform_operation_id.
```

Этот запрос возвращает последнюю версию файла **цепочки импорта** в формате ECI_Import_Chain_File для клиента **микросервера**, обозначенного идентификаторами **operator_id**, **platform_operation_id**.

```
client_exp_auth ::= 'client-exp_auth/' operator_id '/' platform_operation_id.
```

Этот запрос возвращает последнюю версию файла аутентификации экспорта в формате ECI_Export_Authentication_File для клиента **микросервера**, обозначенного идентификаторами **operator_id**, **platform_operation_id**.

9.7.2.3 API соединения экспорта

9.7.2.3.1 Общие сведения

Клиенты ECI могут предоставлять информацию по экспорту **хосту ECI**. Таким образом, **хосту ECI** разрешается связывать систему экспорта с соответствующими **цепочками импорта** от **микросерверов**. **Хост ECI** (и приложение) может определять формирование действующих соединений из всех возможных вариантов. Хост предпринимает попытки соединить **клиентов ECI**, выполняющих экспорт и соответствующий ему импорт, путем отправки выполняющему экспорт **клиенту ECI** запроса на соединение с **цепочкой импорта** для целевого клиента **клиента ECI** импорта. Выполняющий экспорт **клиент ECI**, а также **хост ECI** могут отправлять запрос на отмену соединения или повторное инициирование соединения при наличии обновленных идентификационных данных импорта. Перечень доступных сообщений **соединения экспорта** приводится в таблице 9.7.2.3.1-1.

Таблица 9.7.2.3.1-1 – Сообщения API соединения экспорта

Сообщение	Тип	Направление	Маркер	Описание
reqExpConnNodes	A	H→C	0x0	**Хост ECI** запрашивает у **клиента ECI** дополнительные узлы экспорта
reqExpConnSetup	A	H→C	0x1	**Хост ECI** направляет **клиенту ECI** запрос на инициирование **соединения экспорта** с **клиентом ECI**, выполняющим импорт, на основе **цепочки импорта**
reqExpConnDrop	A	H→C	0x2	**Хосты ECI** отменяют любое предварительно инициированное соединение **клиента ECI**, выполняющего экспорт, и **клиента ECI**, выполняющего импорт
reqExpConnCancel	A	C→H	0x3	**Клиент ECI** завершает все инициированные **соединения экспорта** с **клиентом ECI**, выполняющим импорт
reqExpMhOpen	A	H→C	0x4	**Хост ECI** направляет **клиенту ECI** запрос на создание сеанса экспорта на основе ранее инициализированного **соединения экспорта**
reqExpMhClose	A	H→C	0x5	**Хост ECI** закрывает сеанс экспорта
reqExpMhCancel	A	C→H	0x6	**Клиент ECI** отменяет сеанс экспорта

9.7.2.3.2 Сообщение reqExpConnNodes

H→C reqExpConnNodes() →
C→H resExpConnNodes(ExpConnOption **conn Nodes [])**

- Это сообщение направляет **клиенту ECI** запрос на возврат списка возможных **соединений экспорта**; список возвращается в сообщении-**отклике**. Соответствующие коды ошибок перечислены в таблице 9.7.2.3.2-2.

Определение параметра отклика

connNodes: ExpConn Option[]	В этом списке содержатся учетные данные **ECI** либо третьей стороны, либо **клиентов ECI**, к которым **клиент ECI** может подключиться для выполнения экспорта. Каждый вариант обладает определенным приоритетом: чем выше приоритет, тем ниже вероятность, что экспорт не будет успешно завершен. ExpConnNode определяется в таблице 9.7.2.3.2-1

Таблица 9.7.2.3.2-1 – Определение типа ExpConnNode

```
typedef struct ExpConnNode {
    uint   targetType;
    uint   operatorId;
    uint   targetId;
    uint   targetPriority;
} ExpConnNode;
```

Определения полей

targetType: uint	Тип целевого объекта: Значение, равное 1, соответствует EAC (третья сторона), значение, равное 2, соответствует POC (прямой экспорт). Другие значения не определяются
operatorId: uint	Представление 20-битового идентификатора сертификата **ECI оператора** экспорта целевого объекта: export_authorization_operator_id для целевого объекта EAC и operator_id для целевого объекта POC
targetId: uint	Представление 20-битового идентификатора сертификата **ECI** экспорта export_authorization_id для целевого объекта EAC и platform_operation_id для целевого объекта POC
targetPriority: uint	Приоритет выбора конкретного экспорта состоит из двух слагаемых: • Значение, кратное 1 024, представляющее специальный (коммерческий) приоритет для подключения экспорта к конкретному **микросерверу**. • Значение в диапазоне от 0 до 1 023, представляющее дробь минус 1/1024 от ожидаемых вариантов использования контента, которые могут быть экспортированы посредством данной **микросистемы DRM** экспорта. **Хосты ECI** используют данную информацию для автоматического выбора наиболее подходящей **микросистемы DRM** (при условии, что требования для микроприложения DRM соблюдаются системой с наивысшим приоритетом), и/или для представления вышеуказанной информации как преимущества для **пользователя** при ручном выборе

Таблица 9.7.2.3.2-2 – Коды ошибок reqExpNodeInfo

Название	Описание
ErrExpConnNwAccess	См. таблицу 9.7.2.3.9-1
ErrExpConnAuthProblem	
ErrExpUninitState	

9.7.2.3.3 Сообщение reqExpConnSetup

H→C reqExpConn Setup (CertChainSerial **Import,** CertChainSerial **Auth[],**ushort **connId)** → **C→H resExpConn Setup ()**

- Это сообщение направляет **клиенту ECI** запрос на инициирование (или повторное инициирование) идентификатора connId **соединения экспорта** с **клиентом ECI** при помощи идентификатора **clientId**, используя **цепочку импорта**, цепочки аутентификации экспорта **Auth** и цепочки **клиент ECI–целевой объект**.

Определения параметра запроса

Import: CertChainSerial	**Цепочка импорта** (от сертификата экспорта TPEGC до ESC)
Auth: CertChainSerial[]	Цепочки аутентификации экспорта от корневого объекта до EAC, который выполняет аутентификацию первого сертификата TPEGC в отдельной субцепочке третьей стороны. Цепочки в **Auth** следуют в порядке от соединения TPEGC, выполняющего экспорт, до POC, выполняющего импорт
connId: ushort	Идентификатор **соединения экспорта**, присваиваемый хостом ECI

Определение типа CertChainSerial и типа массива

CertChainSerial – это представление сетевого порядка (с прямым порядком байтов) цепочки ECI_Certificate_Chain, как указывается в таблице 5.4.1-1, дополненное до числа, кратного 32 битам.

CertChainSerial[] определяется следующей структурой данных (quasi-C):

```
typedef struct CertChainSerial {
    uint   numberElements;      /* количество элементов в массиве цепочки*/
    uint   elementIndex[];      /* индекс начала каждого элемента в контейнере данных
                                     chainElements data container */
    uint   chainElements[];  /* контейнер данных с представлениями
                                     SertChainSerial последовательных
                                     цепочек в массиве. */
  } CertChainSerial;
```

Элементы elementIndex и chainElements должны быть представлены массивами встраиваемых данных в структуре данных certChainSerialArray.

Подробная семантика

- Хосты ECI ECI могут создавать запрос reqExpConnSetup от имени существующего соединения с целью информировать клиента ECI, выполняющего экспорт, о (потенциально) новых идентификационных данных импорта клиента ECI, выполняющего импорт. Если текущее соединение нельзя прекратить немедленно, клиентам ECI, выполняющим экспорт, рекомендуется отложить обновление соединения с клиентом ECI, выполняющим импорт, до момента времени, когда будут отсутствовать активные сеансы.

Соответствующие коды ошибок перечисляются в таблице 9.7.2.3.3-1.

Таблица 9.7.2.3.3-1 – Коды ошибок reqExpConnSetup

Название	Описание
ErrExpConnNwAccess	См. таблицу 9.7.2.3.9-1
ErrExpConnAuthProblem	
ErrExpUninitState	
ErrExpInvalidChain	

9.7.2.3.4 Сообщение reqExpConnDrop

H→C reqExpConnDrop(ushort **connId)** →
C→H resExpConnDrop()

- Это сообщение направляет **клиенту ECI** запрос на прекращение **соединения экспорта** с клиентом, идентифицируемым посредством **connId**.

Определения параметра запроса

connId: ushort	Идентификатор соединения экспорта

Предварительные условия (запрос)

1) **Соединение экспорта** (идентифицируемое посредством **connId)** было установлено ранее.

Постусловия (отклик)

1) **Соединение экспорта** (при наличии) закрывается.

Соответствующие коды ошибок перечисляются в таблице 9.7.2.3.4-1.

Таблица 9.7.2.3.4-1 – Коды ошибок reqExpConnDrop

Название	Описание
ErrExpConnNone	См. таблицу 9.7.2.3.9-1

9.7.2.3.5 Сообщение reqExpConnCancel

C→H reqExpConnCancel(ushort **connId)** →
H→C resExpConnCancel()

- Это сообщение информирует **хост ECI** о том, что **соединение экспорта** с идентифицируемое посредством **connId**, завершено **клиентом ECI**.

Определения параметра запроса

connId: ushort	Идентификатор, присвоенный соединению

Предварительные условия (запрос)

1) **Соединение экспорта**, идентифицируемое посредством **connId**, было установлено ранее.

9.7.2.3.6 Сообщение reqExpMhOpen

H→C reqExpMhOpen(ushort **mhExp,** ushort **mhDcr,** ushort **connId) →**
C→H resExpMhOpen(ushort **mhExp)**

- Это сообщение направляет **клиенту ECI** запрос на создание сеанса экспорта, идентифицируемого по **mh указателя медиаданных**, через **соединение экспорта** (идентифицируемое посредством **connId**).

Определения параметра запроса

mhExp: ushort	**Указатель медиаданных, назначаемый хостом ECI** для соединения экспорта
mhDcr: ushort	**Указатель медиаданных** сеанса дешифрования для выполнения экспорта
connId: ushort	Идентификатор, назначаемый **соединению экспорта**

Определения параметра отклика

mhExp: ushort	**Указатель медиаданных,** назначаемый **хостом ECI для** соединения экспорта

Предварительные условия (запрос)

1) **Соединение экспорта** (идентифицируемое посредством **connId**) было установлено ранее.
2) Сеанс дешифрования **mhDcr** был установлен ранее.

Постусловия (запрос)

1) Соединение экспорта устанавливается или произошла ошибка.

Подробная семантика

- Клиент **ECI**, выполняющий экспорт, может приостанавливать и возобновлять экспорт в рамках существующего сеанса, например, на основе включения данного соединения в **группу экспорта**.

Соответствующие коды ошибок перечисляются в таблице 9.7.2.3.6-1.

Таблица 9.7.2.3.6-1 – Коды ошибок reqExpMhOpen

Название	Описание
ErrExpConnNone	См. таблицу 9.7.2.3.9-1
ErrExpDcrMhNone	

9.7.2.3.7 Сообщение reqExpMhClose

H→C reqExpMhClose(ushort **mhExp) →**
C→H resExpMhClose(ushort **mhExp)**

- Это сообщение направляет **клиенту ECI** запрос на закрытие сеанса экспорта, идентифицируемого по **mh указателя медиаданных**, через **соединение экспорта** (идентифицируемое посредством **connId**).

Определения параметра запроса

mhExp: ushort	**Указатель медиаданных,** назначаемый **хостом ECI** для **соединения экспорта**

Определения параметра отклика

mhExp: ushort	**Указатель медиаданных,** назначаемый **хостом ECI** для **соединения экспорта**

Предварительные условия (запрос)

1) Сеанс экспорта **mhExp** установлен ранее и еще не завершен.

Постусловия (запрос)

1) Сеанс экспорта **mhExp** остановлен.

Соответствующие коды ошибок перечисляется в таблице 9.7.2.3.7-1.

Таблица 9.7.2.3.7-1 – Коды ошибок reqExpMhClose

Название	Описание
ErrExpMhNone	См. таблицу 9.7.2.3.9-1

9.7.2.3.8 Сообщение reqExpMhCancel

C→H reqExpMhCancel(ushort **mhExp) →**
H→C resExpMhCancel(ushort **mhExp)**

- Это сообщение информирует **хост ECI** о том, что **клиент ECI** остановил сеанс экспорта **mhExp**.

Определения параметра запроса

mhExp: ushort	**Указатель медиаданных** назначен **хостом ECI** для **соединения экспорта**

Определения параметра отклика

mhExp: ushort	**Указатель медиаданных** назначен **хостом ECI** для **соединения экспорта**

Предварительные условия (запрос)

1) Сеанс экспорта **mhExp** был установлен ранее.

2) **Клиент ECI** завершил сеанс.

9.7.2.3.9 Коды ошибок для API соединения экспорта

Значения ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, определены в таблице 9.7.2.3.9-1.

Таблица 9.7.2.3.9-1 – Коды ошибок API медиасеанса для медиаданных транспортных потоков

Название	Значение	Описание
ErrExpConnNwAccess	–256	Доступ к сети, предоставляющей информацию по запросу, оказывается невозможным либо неожиданно замедляется и не может быть завершен
ErrErrConnAuthProblem	–257	Обнаружены внутренние несоответствия в предоставленных данных, которые не позволяют завершить запрос
ErrEcxpConnUninitState	–258	Сначала **клиент ECI** требует предоставления и/или выполнения других функций, что позволяет отправить отклик на этот запрос
ErrExpConnInvalidChain	–259	Цепочка, предоставленная **клиенту ECI**, была признана недействительной, и/или ее аутентификация с использованием цепочек аутентификации была невозможна
ErrExpConnNone	–260	Соединение не существовало
ErrExpMhNone	–261	Сеанс экспорта, обозначенный **указателем медиаданных**, не поддерживается **клиентом ECI**
ErrExpDcrMhNone	–262	**Клиент ECI** не поддерживает сеанс дешифрования, обозначенный **указателем медиаданных**

9.7.2.4 API соединения импорта

9.7.2.4.1 Общие сведения

Клиенты ECI могут предоставлять свои **цепочки импорта хосту ECI**. Это позволяет **хосту ECI** подключать **клиента ECI**, выполняющего импорт, к соответствующим вариантам экспорта **микросерверов**. **Хост ECI** и приложение могут выбирать для установки соединение (соединения), которое формируется на основе доступных вариантов. **Хост ECI** может начать установку соединения между **клиентами ECI**, выполняющими экспорт и выполняющими импорт, предварительно запросив

разрешение у клиента, выполняющего импорт, на подключение к **клиенту ECI**, выполняющему экспорт. Клиент, выполняющий импорт, может отказаться от такого соединения, например, по коммерческим соображениям, связанным с оператором. Если соединение установлено, **клиент ECI**, выполняющий импорт, а также **хост ECI** могут отправлять запрос на отмену соединения или повторную инициализацию соединения в случае наличия обновленных идентификационных данных импорта.

Цепочки ввода данных идентифицируются по первому узлу, то есть по идентификаторам **ECI** EAOC и EAC для TPEGC. В приведенной ниже таблице 9.7.2.4.1-1 этот объект обозначается как *узел импорта*.

Таблица 9.7.2.4.1-1 – Сообщения API соединения импорта

Сообщение	Тип	Направление	Маркер	Описание
reqImpConnNodes	A	H→C	0x0	**Хост ECI** направляет **клиенту ECI**, выполняющему импорт, запрос на предоставление узлов импорта
reqImpConnChain	A	H→C	0x1	**Хост ECI** направляет **клиенту ECI**, выполняющему импорт, запрос на предоставление цепочки ввода данных для конкретного узла импорта
reqImpConnChainRenew	A	C→H	0x2	**Клиент ECI** направляет **хосту ECI** запрос на повторную инициализацию соединения с использованием обновленной **цепочки импорта**
reqImpConnSetup	A	H→C	0x3	**Хост ECI** направляет **клиенту ECI**, выполняющему импорт, запрос на инициализацию **соединения импорта** с конкретным **клиентом ECI**, выполняющим экспорт, через узел импорта
reqImpConnDrop	A	H→C	0x4	**Хост ECI** прекращает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт
reqImpConnCancel	A	C→H	0x5	**Клиент ECI** завершает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт

9.7.2.4.2 Сообщение reqImpConnNodes

H→C reqImpConnNodes () →
C→H resImpConnNodes(ImpConnNode **nodes[])**

- Это сообщение позволяет **хосту ECI** направлять **клиенту ECI**, выполняющему импорт, запрос на предоставление узлов импорта.

Определения параметра отклика

nodes[]: ImpConnNode	Массив узлов импорта и количество посредников (третьих сторон). Структура ImpConnNodes определена в таблице 9.7.2.4.2-1

Таблица 9.7.2.4.2-1 – Определение типа ImpConnOption

```
typedef str uct ImpConnNode {
    uint   targetType;
    uint   operatorId;
    uint   targetId;
    uint   intermediaries
} ImpConnNode;
```

Определения полей

targetType: uint	Тип целевого объекта: 1-EAC (третья сторона), 2-POC (прямой экспорт). Остальные значения не определены
operatorId: uint	Представление 20-битового идентификатора сертификата **ECI оператора** импорта **целевого объекта**: export_authorization_operator_id для целевого объекта EAC или operator_id для целевого объекта POC
targetId: uint	Представление 20-битного идентификатора сертификата **ECI** импорта **целевого объекта**: export_authorization_id для целевого объекта EAC или platform_operation_id для целевого объекта POC
intermediaries: uint	Представляет количество промежуточных третьих сторон от входного узла до POC **клиента ECI**, выполняющего импорт. **Хосты ECI** выбирают наиболее короткую **цепочку импорта** среди альтернативных вариантов для параметров экспорта, которые имеют тот же targetPriority для **клиента ECI**, выполняющего экспорт

Соответствующие коды ошибок перечислены в таблице 9.7.2.4.2-2.

Таблица 9.7.2.4.2-2 – Коды ошибок reqExpConnInfo

Название	Описание
ErrImpConnNwAccess	См. таблицу 9.7.2.4.7-1
ErrImpConnAuthProblem	
ErrImpUninitState	

9.7.2.4.3 Сообщения reqImpConnChain и reqImpConnChainRenew

H→C reqImpConnChain(ImpConnNode **node)** →
C→H resImpConnChain(CertChainSerial **Import,** CertChainSerial **Auth[])**

- Это сообщение позволяет **хосту ECI** направлять **клиенту ECI**, выполняющему импорт, запрос на предоставление цепочки ввода данных для конкретного узла импорта.

C→H reqImpConnChainRenew(CertChainSerial **Import,** CertChainSerial**Auth[])** →
H→C resImpConnChainRenew()

- Это сообщение позволяет **клиенту ECI** направлять **хосту ECI** запрос на повторную инициализацию соединения с использованием обновленной **цепочки импорта**.

Параметр запроса для reqImpConnChain

node: ImpConnNode	Узел импорта, для которого **цепочка импорта** должна возвращаться **хосту ECI**

Определения параметра запроса для reqImpConnChainRenew
и Определения параметра отклика для reqImpConnChain

Import: CertChainSerial	**Цепочка импорта** (от экспорта TPEGC до ESC).
Auth: CertChainSerial[]	Цепочки аутентификации экспорта от корневого сертификата до EAC, который аутентифицирует первый TPEGC в одиночной субцепочке третьих лиц. Цепочки в **Auth** располагаются по порядку, начиная от TPEGC, выполняющего экспорт, до POC, выполняющего импорт.

Начальные условия reqImpConnChainRenew (запрос)

1) **Соединение импорта** с **клиентом ECI** было установлено ранее с использованием элемента в предоставленной цепочке.

Подробная семантика для reqImpConnChainRenew

- **Хост** ECI должен незамедлительно передавать обновленную информацию о цепочке соответствующим **клиентам ECI**, выполняющим экспорт.
- Рекомендуется, чтобы операторы предоставляли обновленные цепочки заранее, до того как предыдущая цепочка будет признана устаревшей, что гарантирует бесперебойное оказание услуг.

Коды ошибок, связанных с reqImpConnChain, перечислены в таблице 9.7.2.4.3-1.

Таблица 9.7.2.4.3-1 – Коды ошибок reqImpConnChain

Название	Описание
ErrImpConnNwAccess	См. таблицу 9.7.2.4.7-1.
ErrImpConnAuthProblem	
ErrImpConnUninitState	

Коды ошибок, связанных с reqImpConnChainRenew, перечислены в таблице 9.7.2.4.3-2.

Таблица 9.7.2.4.3-2 – Коды ошибок reqImpConnChainRenew

Название	Описание
ErrImpConnNoConn	См. таблицу 9.7.2.4.7-1

9.7.2.4.4 Сообщение reqImpConnSetup message

H→C reqImpConnStart (ImpConnNode **node,** ushort **exportClientId,** ushort **connId)** →
**C→H resImpConnStart(**)

- Это сообщение позволяет **хосту ECI** направлять **клиенту ECI**, выполняющему импорт, запрос на установку **соединения импорта** с конкретным **клиентом ECI**, выполняющим экспорт, через узел импорта.

Параметры запроса

node: ImpConnNode	Узел импорта, через который установлено соединение
exportClientId: ushort	Идентификация **хостом ECI клиента ECI**, выполняющего экспорт
connId: ushort	**Идентификатор, присвоенный соединению импорта**

Подробная семантика

- **Клиент ECI** может отклонять **соединение импорта** по коммерческим соображениям, связанным с оператором.

Соответствующие коды ошибок перечислены в таблице 9.7.2.4.4-1.

Таблица 9.7.2.4.4-1 – Коды о шибок reqExpConnStart

Название	Описание
ErrImpConnNwAccess	См. таблицу 9.7.2.4.7-1
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnRefuseComm	
ErrImpConnUnknError	

9.7.2.4.5 Сообщение reqImpConnDrop message

H→C reqImpConnDrop (ushort **connId)** →
C→H resImpConnDrop()

- Это сообщение позволяет **хосту ECI** прекращать **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт.

Параметры запроса

connId: ushort	Идентификация **хостом ECI соединения импорта**, подлежащего прекращению

Предварительные условия (запрос)

1) Ранее инициализированное **соединение импорта** (с идентификатором **connId**).

Постусловия (отклик)

1) **Соединение экспорта** (при наличии) закрыто.

Соответствующие коды ошибок перечислены в таблице 9.7.2.4.5-1.

Таблица 9.7.2.4.5-1 – Коды ошибок reqExpConnInfo

Название	Описание
ErrImpConnNwAccess	См. таблицу 9.7.2.4.7-1
ErrImpConnAuthProblem	
ErrImpConnUninitState	
ErrImpConnNoConn	

9.7.2.4.6 Сообщение reqImpConnCancel

C→H reqImpConnCancel (ushort **connId)** →
H→C resImpConnCancel()

- Это сообщение позволяет **клиенту ECI** завершать **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт.

Параметры запроса

connId: ushort	Ранее инициализированное **соединение импорта** (определяется по connId)

Предварительные условия (запрос)

1) **Соединение импорта** было установлено ранее с клиентом, имеющим идентификатор клиента **ECI Host exportClientId,** и закрыто.

9.7.2.4.7 Коды ошибок для API соединения экспорта

Значения ошибок, связанных с интерфейсом API, которые могут возвращаться в сообщениях-**откликах** для данного API, перечислены в таблице 9.7.2.4.7-1.

Таблица 9.7.2.4.7-1 – Коды ошибок API медиасеанса для медиаданных транспортных потоков

Название	Значение	Описание
ErrImpConnNwAccess	–256	Доступ к сети, предоставляющей запрошенную информацию, неожиданно замедлился
ErrImpConnAuthProblem	–257	Обнаружены внутренние несоответствия в предоставленных данных, которые не позволяют завершить запрос
ErrImpUninitState	–258	Сначала **клиент ECI** требует предоставления и/или выполнения других функций, что позволяет отправить отклик на этот запрос
ErrImpConnRefuseComm	–259	Цепочка, предоставленная **клиенту ECI**, была признана недействительной и/или ее аутентификация с использованием цепочек аутентификации была невозможна
ErrImpConnRefuseComm	–260	**Клиент ECI**, выполняющий импорт, отклоняет подключение к **клиенту ECI**, выполняющему экспорт, по коммерческим соображениям
ErrImpConnUnknError	–261	**Клиент ECI**, выполняющий импорт, обнаружил неизвестную ошибку
ErrExpConnNone	–262	Соединение не существовало

9.7.2.5 API повторного шифрования

9.7.2.5.1 Общие сведения

API повторного шифрования позволяет **микросерверу** повторно шифровать контент из **соединения импорта**, определенного для одного клиента из группы, в целях последующего декодирования **микроклиентом**. При потоковой передаче может существовать необходимость почти мгновенного выполнения декодирования. Повторное воспроизведение в последующем сеансе может быть не разрешено. Как вариант, повторно зашифрованный контент может быть сохранен или сдвинут во времени; при этом **микроклиенту**, выполняющему декодирование, предоставляется соответствующая информация для дешифрования, что позволяет выполнить декодирование контента **микроклиентом** позже.

На этапе обнаружения приложению разрешено соотносить **микросервер** с возможным **целевым объектом** (**микроклиентом** или группой **микроклиентов**) и обмениваться необходимой информацией по аутентификации между **микроклиентом** и **микросервером**, что позволяет выполнять аутентификацию **микроклиента** и закладывает основу для доверительного обмена контентом. **Хост ECI** может выбрать режим двунаправленной связи (IP-соединение или передачу сообщения через **хост ECI**), что позволяет поддерживать более совершенные протоколы аутентификации между **микросервером** и **микроклиентом**.

На основе соединения для повторного шифрования с **целевым объектом** и **соединения импорта хост ECI** создает сеанс **указателя медиаданных** в режиме повторного шифрования, синхронизации и формата данных, который требуется приложению и может поддерживаться **микросервером**.

Как только соединение для повторного шифрования установлено, **хост ECI** может создать сеанс **указателя медиаданных** с **микросервером** и начинать повторное шифрование контента из установленного **соединения импорта** для **целевого объекта** (**клиента ECI** или группы **клиентов ECI**). Могут быть созданы несколько одновременных операций повторного шифрования одного и того же контента; при этом каждый раз используется собственный сеанс **указателя медиаданных**. **Хост**

ECI отвечает за то, чтобы контент для сеанса **указателя медиаданных** для повторного шифрования был получен из аутентифицированного **указателя медиаданных** экспорта при установлении **соединения экспорта**. Неавторизованное ошибочное соединение может привести к сбою аутентификации экспорта.

Слова управления повторным шифрованием применяются к импортируемому дешифрованному контенту, а новые указатели (URI и т. д.) применяются для повторно зашифрованного контента при использовании системы AS.

Далее описываются три основных *режима шифрования*:

1) Режим потоковой онлайн-передачи: и **микросервер**, и **микроклиент** активны в одно и то же время. Они обмениваются сообщениями напрямую (через IP-соединения) или точными сообщениями через соответствующие **хосты ECI**.

2) Режим потоковой офлайн-передачи: **микросервер** шифрует контент "на лету" и непрерывно выдает новые данные, необходимые **микроклиенту** для дешифрования. Результат может быть предоставлен позже (режим смещения во времени) или сохранен.

3) Режим офлайн-хранения: **микросервер** шифрует контент и по завершении генерирует данные, необходимые **микроклиенту** в начале декодирования контента.

На рисунке 9.7.2.5.1-1 представлена общая схема различных режимов шифрования.

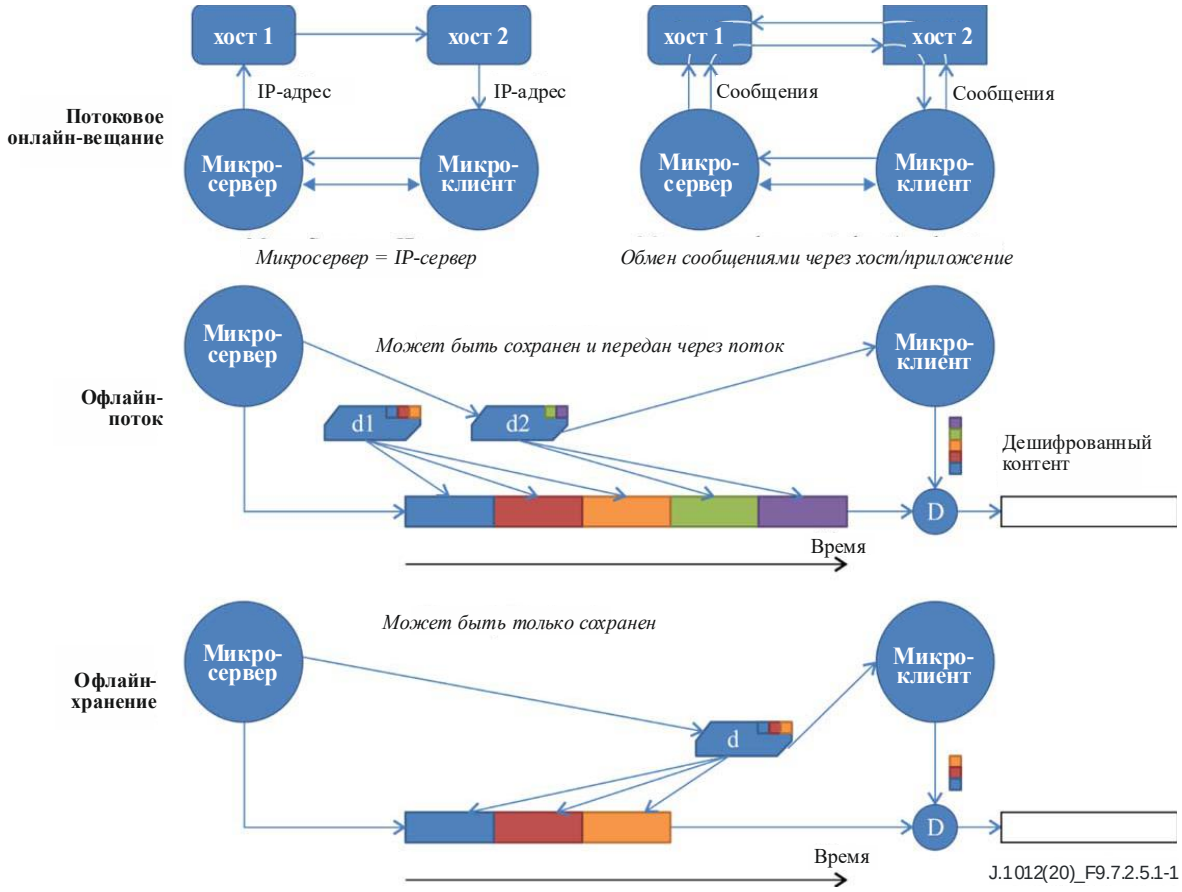


Рисунок 9.7.2.5.1-1 – Режимы шифрования для сеансов микросистем DRM

Данные, необходимые для дешифрования контента, подлежащего обмену между **микросервером** и **микроклиентом** в двух офлайновых режимах шифрования, могут передаваться в следующих *режимах формата данных*.

4) Общий режим: **микросервер** генерирует контейнеры скрытых данных, содержащие информацию, необходимую для дешифрования контента **микроклиентом**.

5) Режим ISOBMFF (только для режима операций с файлами, идентичного *режиму синхронизации*): **микросервер** генерирует блоки PSSH для включения в файл ISOBMFF

[ISO/EC 14496-12]. **Хост ECI** может создавать файлы ISOBMFF, путем надлежащего включения блоков PSSH в блоки ISOBMFF MOOV или MOOF.

В *режиме синхронизации* поддерживаются два механизма, позволяющие связать корректное слово управления с секцией контента, применимой ко всем вышеперечисленным режимам повторного шифрования.

6) В режиме транспортного потока (режиме чередования битов) **микросервер** генерирует секции ECM, которые могут быть пакетированы и помещены в транспортный поток **хостом ECI**. Сообщение ECM вводится перед криптопериодом, для которого оно предоставляет информацию, позволяющую рассчитать слово управления.

7) В режиме операций с файлами **микросервер** создает зашифрованные слова управления, на которые ссылаются явные идентификаторы KeyID, представленные в дополнительной информации для дешифрования. **Хост ECI** сохраняет связь идентификаторов KeyID с секцией контента, зашифрованной определенным словом управления, таким образом чтобы **микроклиент** мог сформировать корректное слово управления для дескремблирования.

В офлайновом режиме осуществляется синхронизация дополнительных данных, необходимых для дешифрования или вычисления KeyId или сообщений ECM и явно указывающих на временну̀ю зависимость данных от KeyId или номера ECM.

Не требуется, чтобы все **микросерверы** поддерживали все режимы работы. Во время инициализации, непосредственно после использования API обнаружения **микросервер** передает информацию о режимах (комбинация режимов шифрования, формата данных и синхронизации), которые он может поддерживать.

Как только сеанс **указателя медиаданных** создан, он может быть запущен и остановлен **хостом ECI** и отменен **клиентом ECI**.

Сообщения для API повторного шифрования перечислены в таблице 9.7.2.5.1-1.

Таблица 9.7.2.5.1-1 – Сообщения API повторного шифрования

Сообщение	Тип	Направление	Маркер	Описание
setEncrModes	set	C→H	0x0	**Микросервер** информирует **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации)
reqEncrTargets	A	H→C	0x1	**Хост ECI** направляет **микросерверу** запрос на предоставление **целевых** узлов, которые он может аутентифицировать для дешифрования
reqEncrConnSetup	A	H→C	0x2	**Хост ECI** направляет **клиенту ECI запрос** на создание **соединения целевого объекта** для повторного шифрования и предварительную аутентификацию **целевого объекта** повторного шифрования для последующей ссылки при настройке сеанса **указателя медиаданных**
reqEncrConnDrop	A	H→C	0x3	**Хост ECI** направляет **клиенту ECI** запрос на исключение любой информации по ранее аутентифицированному соединению повторного шифрования
reqEncrConnCancel	A	C→H	0x4	**Клиент ECI** отменяет ранее установленное соединение **целевого объекта** шифрования.
reqEncrMhOpen	A	H→C	0x5	**Хост ECI** направляет **клиенту ECI** запрос на открытие сеанса **указателя медиаданных** для повторного шифрования контента из входящего **соединения импорта** для установленного соединения повторного шифрования
reqEncrMhClose	A	H→C	0x6	**Хост ECI** закрывает **сеанс повторного шифрования** с **клиентом ECI**
reqEncrMhCancel	A	C→H	0x7	**Клиент ECI** завершает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт
reqEncrMhStart	A	H→C	0x8	**Хост ECI** направляет **клиенту ECI запрос** на запуск операции повторного шифрования для сеанса **указателя медиаданных**
reqEncrMhStop	A	H→C	0x9	**Хост ECI** направляет **клиенту ECI** запрос на остановку операции повторного шифрования для сеанса **указателя медиаданных**
reqEncrMhQuit	A	C→H	0xA	**Клиент ECI** информирует **хост ECI** о том, что операция повторного шифрования **указателя медиаданных** прекращена

Таблица 9.7.2.5.1-1 – Сообщения API повторного шифрования

reqEncrIpServer	A	H→C	0xB	**Хост ECI** запрашивает адрес IP-сервера **микросервера,** для того чтобы разрешить **микроклиентам** создание IP-соединений
reqEncrMsgSend	A	C→H	0xC	**Микросервер** направляет **хосту ECI** запрос на переадресацию сообщения **целевому объекту** относительно сеанса **указателя медиаданных**
reqEncrMsgRecv	A	H→C	0xC	**Хост ECI** передает **микросерверу** сообщение от **целевого объекта** относительно сеанса **указателя медиаданных**
reqEncrTsData	A	C→H	0xE	**Микросервер** предоставляет **хосту ECI** данные для переадресации **целевому микроклиенту указателя медианных** для дешифрования, включая информацию синхронизации для сообщений ECM
reqEncrTsEcm	A	C→H	0xF	**Микросервер** предоставляет секцию ECM, которую **микроклиент** запрашивает для дешифрования в следующем криптопериоде
reqEncrFileData	A	C→H	0x10	**Микросервер** предоставляет **хосту ECI** сообщение для переадресации **целевому микроклиенту указателя медианных** для дешифрования, включая информацию синхронизации для KeyID

9.7.2.5.2 Сообщение setEncrModes

C→H setEncrModes(EciEncrModes **modes)**

- Это сообщение позволяет **микросерверу** информировать **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации).

Определения параметра запроса

modes: EciEncrModes	Режимы шифрования, поддерживаемые **микросервером**. Тип EciEncrModes определен в таблице 9.7.2.5.2-1.

Таблица 9.7.2.5.2-1 – Определение типа EciEncrModes

```
Typedef uint EciEncrModes;
```

Определения битов

Название	Бит	Поддержка режима микросервера при значении, равном 0b1
OnlineIpMode	0	Поддерживает онлайн-режим IP
OnlineMsgMode	1	Поддерживает режим онлайн-сообщений
OfflineStreamMode	2	Поддерживает режим потоковой офлайновой передачи
OfflineStorageMode	3	Поддерживает режим офлайн-хранения
OfflineDataMode	4	Поддерживает по умолчанию контейнеры формата данных для дешифрования данных в офлайновом режиме. Не применяется, если не выбран офлайновый режим
OfflineIsobmffMode	5	Поддерживает блоки PSSH формата ISOBMFF для дешифрования данных в офлайновом режиме. Не применяется, если не выбран офлайновый режим
SyncTs	6	Синхронизирует слова управления с криптопериодами для контента, разделенными чередующимися битами в формате транспортного потока
SyncFile	7	Синхронизируется по файловым форматам с использованием идентификации KeyID для связи секций контента с соответствующим контрольным словом
прочее	RFU	Зарезервировано для использования в будущем

9.7.2.5.3 Сообщение reqEncrTargets

H→C reqEncrTargets() →
C→H resEncrTargets(EncrTarget **target[])**

- Это сообщение позволяет **хосту ECI** направлять **микросерверу** запрос на предоставление целевых объектов шифрования, которые он может аутентифицировать.

Определения параметра отклика

target: EncrTarget[]	Список целевых объектов шифрования, которые может аутентифицировать **микросервер**. Определение типа TargetClient приведено в таблице 9.7.2.5.3-1

Таблица 9.7.2.5.3-1 – Определение типа EncrTarget

```
typedef struct EncrTarget {
    uint   targetType;
    byte   target[8];
} EncrTarget;
```

Определения полей

targetType: uint	Тип целевого объекта шифрования: значение, равное 1, соответствует отдельному клиенту, значение, равное 2, соответствует группе клиентов; остальные значения зарезервированы для использования в будущем
target: byte[8]	Идентификатор, представляющий целевой объект: Значение определено в рамках **микросистемы DRM**. Соответствие **хосту ECI** определяется с точки зрения равенства полей **targetType** и **target**

Подробная семантика

- **Хост ECI** может соответствовать потенциальным **целевым микроклиентам** на основе **целевого объекта**. Определение расположения потенциальных **микроклиентов** производится на усмотрение приложения и/или **хоста ECI.**
- **Хосты ECI**, которые намереваются выполнять функции локального PVR и смещения во времени (используя либо интегрированный, либо подключенный/сетевой накопитель, на котором они могут хранить зашифрованный контент и связанные с ним данные), могут предпринять попытку сопоставить **микросервер**, который способен работать в режиме **OfflineStreamMode**, с **микроклиентами**, установленными на том же **хосте ECI**.

9.7.2.5.4 Сообщение reqEncrConnSetup

H→C reqEncrConnSetup(ushort **targetConnId**, EciEncrTarget **target**, ushort **credLen**, byte **cred[])**
C→H resEncrConnSetup(ushort **targetConnId)**

- Это сообщение позволяет **хосту ECI** направлять **микросерверу** запрос на создание соединения для повторного шифрования с **целевым объектом** и (предварительную) аутентификацию **целевого объекта**. Коды ошибок определены в таблице 9.7.2.5.19-1.

Определения параметра запроса

targetConnId: ushort	Идентификатор для дополнительных ссылок на **целевой объект** между **хостом ECI** и микросервером
target: EciEncrTarget	Идентификатор, представляющий **целевой объект** для аутентификации. Значение определено в рамках **микросистемы DRM**. Соответствие **хосту ECI** определяется с точки зрения равенства полей **targetType** и **target**
credLen: ushort	Длина параметра cred в байтах
cred: byte[]	Информация по учетным данным, полученным от **целевого объекта**, подлежащего аутентификации **микросервером**

Определения параметра отклика

targetConnId: ushort	Идентификатор для дополнительных ссылок на **целевой объект** между **хостом ECI** и **микросервером**

Подробная семантика

- Если **targetConnId** равен **targetConnId**, который использовался ранее **хостом ECI**, но не был удален впоследствии, можно предположить, что предыдущий **целевой объект**, связанный с targetConnId, заменен или обновлен.

Предварительные условия (запрос)

1) **Целевой объект** должен быть равен **целевому объекту**, ранее предоставленному **хосту ECI микросервером** в сообщении **resEncrTargets**. В противном случае для данного параметра возвращается сообщение об ошибке.

2) **Целевой объект** должен соответствовать **целевому объекту** предоставленному **микроклиентом**, и разрешать выполнение аутентификации с использованием **cred**.

Постусловия (отклик)

1) Возвращается статус аутентификации. Следует отметить, что результат не всегда является окончательным и может, например, предоставлять неверные идентификационные данные, в результате чего зашифрованный контент не может быть декодирован.

2) **Хост ECI** может ссылаться на (предварительно) аутентифицированный **целевой объект** через **targetConnId**.

Таблица 9.7.2.5.4-1 – Коды ошибок reqEncrConnSetup

Название	Описание
ErrEncrAuthFail	См. таблицу 9.7.2.5.19-1
ErrEncrAuthInconclusive	

9.7.2.5.5 Сообщение reqEncrConnDrop

H→C reqEncrConnDrop(ushort **targetConnId) →**
C→H resEncrConnDrop(ushort **targetConnId)**

- Это сообщение позволяет **хосту ECI** направлять **микросерверу** запрос на исключение любой информации по соединению повторного шифрования, ранее прошедшему предварительную аутентификацию.

Определения параметра запроса

targetConnId: ushort	Идентификатор соединения **целевого объекта**, которое удаляется **микросервером**

Определения параметра отклика

targetConnId: ushort	Идентификатор соединения **целевого объекта**, удаленного с **микросервера**

Предварительные условия (запрос)

1) **TargetConnId** должен существовать на **микросервере**.

Предварительные условия (отклик)

1) **Микросервер** больше не связывает **targetConnId** с предварительно аутентифицированным соединением **целевого объекта** и освободил все ресурсы, связанные с предварительной аутентификацией **targetConnId**.

9.7.2.5.6 Сообщение reqEncrConnCancel

C→H reqEncrConnCancel(ushort **targetConnId) →**
H→C resEncrConnDrop(ushort **targetConnId)**

- Это сообщение позволяет **микросерверу** информировать **хост ECI** об отмене соединения повторного шифрования, ранее прошедшего предварительную аутентификацию.

Определения параметра запроса

targetConnId: ushort	Идентификатор соединения целевого объекта, отмененного **микросервером**

Определения параметра отклика

targetConnId: ushort	Идентификатор соединения целевого объекта, отмененного **микросервером**

Предварительные условия (запрос)

1) **TargetConnId** должен существовать на **микросервере**.

Предварительные условия (отклик)

1) Значение TargetConnId свободно и может быть переназначено **хостом ECI** как часть последующего сообщения reqEcnrConnSetup.

9.7.2.5.7 Сообщение reqEncrMhOpen

H→C reqEncrMhOpen(ushort **mh,** ushort **impConn,** ushort **targetConnId,** EncrMode **mode) →**
C→H resEncrMhOpen(ushort **mh)**

- Это сообщение позволяет **хосту ECI** направлять **клиенту ECI** запрос на открытие сеанса **указателя медиаданных** для повторного шифрования контента под управлением **микросервера** из входящего **соединения импорта** для переадресации предварительно аутентифицированного целевого объекта. Коды ошибок определены в таблице 9.7.2.5.7-1.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса шифрования, подлежащего открытию, распределенный **хостом ECI**
impConn: ushort	Идентификатор входного соединения, через которое получен контент, подлежащий повторному шифрованию
targetConnId: ushort	Идентификатор соединения **целевого объекта**, через которое получен контент, подлежащий повторному шифрованию
mode: EncrMode	Спецификация отдельного режима (режим шифрования, режим формата данных, режим синхронизации) для функционирования **микросервера**, выбранная из режимов, поддерживаемых **микросервером**, как указано параметром **setEncrModes**

Определение параметра отклика

mh: ushort	**Указатель медиаданных сеанса шифрования, подлежащего открытию, распределенный хостом ECI**

Предварительные условия (запрос)

1) **Хост ECI** зарезервировал все ресурсы, необходимые для создаваемого сеанса.

2) **impConn** и **targetConnId** устанавливаются **хостом ECI** и **микросервером**.

Предварительные условия (отклик)

1) При положительном результате **микросервер** резервирует все ресурсы, необходимые в большинстве случаев для повторного шифрования контента в запрашиваемом сеансе. Должен также быть включен доступ к любым внешним ресурсам (серверы DRM, **смарт-карты** и т. д.), которые в большинстве случаев требуются для выполнения дешифрования.

ПРИМЕЧАНИЕ. – Ресурсы, требуемые в виде исключения, или ресурсы, доступные в большинстве случаев, исключаются по мере необходимости.

2) При возврате **ErrDcrUserDelay микросервер** ожидает введения данных **пользователем** для открытия сеанса (например, для получения доступа к **смарт-карте** или аутентификации **пользователем**). **Хост ECI** может повторять отправку запроса **reqEncrMhOpen** (с такими же параметрами) до возврата положительного результата либо точно определенной ошибки; как вариант, он может отправлять **reqEncrMhClose** для завершения ожидающего сеанса. **Микросервер** может отменять сеанс, отправив **reqDcrMhCancel** в том случае, если он не может получить необходимые данные, введенные **пользователем**.

Таблица 9.7.2.5.7-1 – Коды ошибок reqEncrMhOpen

Название	Описание
ErrEncrUserMissing	См. таблицу 9.7.2.5.19-1
ErrEncrCardMissing	
EnrEncrServiceMissing	
ErrEncrResourceMissing	
ErrEncrMmiMissing	
ErrEncrClientAuthError	

9.7.2.5.8 Сообщение reqEncrMhClose

H→C reqEncrMhClose(ushort **mh) →**
C→H resEncrMhClose(ushort **mh)**

- Это сообщение позволяет хосту ECI закрывать сеанс повторного шифрования с микросервером.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса шифрования, подлежащего закрытию

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса шифрования, подлежащего закрытию

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** находится в открытом состоянии (или произойдет ошибка).

Предварительные условия (отклик)

1) Освобождаются ресурсы, необходимые **микросерверу** для поддержания сеанса.
2) Состояние **mh** закрыто клиентом.

9.7.2.5.9 Сообщение reqEncrMhCancel

C→H reqEncrMhCancel(ushort **mh,** uchar **reason) →**
H→C resEncrMhCancel(ushort **mh)**

- Это сообщение позволяет **клиенту ECI** закрывать **сеанс повторного шифрования** с заданным **клиентом ECI,** выполняющим экспорт (**микросервером)**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса шифрования, который был отменен **микросервером**
reason: uchar	Обоснования отмены сеанса дешифрования. Значения определены в таблице 9.7.2.5.9-1

Таблица 9.7.2.5.9-1 – Значения для обоснования reqEncrMhCancel

Название	Значение	Описание
EncrMhUndefined	0x00	Произошла неизвестная ошибка, и **микросервер** вынужден отменить сеанс
EncrMhCardMissing	0x01	**Смарт-карта**, требуемая для повторного шифрования, не может быть подключена (повторно подключена) и использована для повторного шифрования контента в течение приемлемого временного интервала
EncrMhServiceMissing	0x02	Служба, не относящаяся к **CPE** и поддерживающая предоставление **микросервером** услуг шифрования, необходимых для проведения сеанса дешифрования, недоступна в течение приемлемого временного интервала
EncrMhResourceMissing	0x03	Ресурс, относящийся к **CPE** и необходимый для предоставления услуг шифрования, недоступен для **микросервера** в течение приемлемого временного интервала (исключая **DcrMhMmiMissing**)
EncrMhMmiMissing	0x04	**Микросерверу** не удалось получить ресурс сеанса MMI для взаимодействия с **пользователем**, необходимого для проведения **сеанса повторного шифрования**, в течение приемлемого временного интервала
RFU	Other	Зарезервировано для использования в будущем

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса шифрования, который был отменен

Предварительные условия (запрос)

1) **Клиент ECI** освобождает любые ресурсы, которые ему нужны специально для сеанса.

Постусловия (запрос)

1) **Хост ECI** может освободить любые ресурсы, относящиеся к **указателю медиаданных**.

Постусловия (отклик)

1) Сеанс **указателя медиаданных** закрыт **хостом ECI**.

9.7.2.5.10 Сообщение reqEncrMhStart

H→C reqEncrMhStart(ushort **mh) →**
C→H resEncrMhStart(ushort **mh)**

- Это сообщение позволяет **хосту ECI** направлять **клиенту ECI** запрос на запуск операции повторного шифрования для сеанса **указателя медиаданных**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса шифрования для запуска

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса шифрования, который был запущен

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** в открытом состоянии (или произойдет ошибка).

Предварительные условия (отклик)

1) Сеанс **указателя медиаданных** начат (или произошла ошибка).

Подробная семантика

- Шифрование контента продолжается, так как контент предоставлен **клиентом ECI**, выполняющим экспорт.
- Любые конфликты или ошибки URI **клиента ECI**, выполняющего экспорт, при аутентификации **микросервера** для экспорта контента не позволят получить зашифрованный контент. Статус управления выходными данными URI **микросервера** OcAnyOther задан равным 0b1; все остальные биты управления выходными данными должны быть заданы равными 0b0 (это означает, что вывод данных не разрешен). **Микросервер** предпринимает попытки шифрования контента до тех пор, пока не получит разрешение.
- Любые сообщения инициализации **микроклиента** доступны через соответствующие этой цели сообщения. Для сеансов с режимом повторного шифрования **OfflineStreamMode** первые данные инициализации для дешифрования контента формируются вскоре после сообщения **resEncrMhStart**.
- Отправка второго сообщения **reqEncrMhStart** перед завершением процесса шифрования завершает предыдущий процесс и запускает следующий.

9.7.2.5.11 Сообщение reqEncrMhStop

H→C reqEncrMhStop(ushort **mh) →**
C→H resEncrMhStop(ushort **mh)**

- Это сообщение позволяет **хосту ECI** направлять **микросерверу** запрос на остановку операции повторного шифрования для сеанса **указателя медиаданных**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для завершающегося сеанса шифрования

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для завершенного сеанса шифрования

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** находится в запущенном состоянии (или произойдет ошибка).

Предварительные условия (отклик)

1) Сеанс указателя медиаданных завершен.

Постусловия (отклик)

1) Значение сеанса **указателя медиаданных** может быть повторно использовано **хостом ECI**.

Подробная семантика

- На сеансах с режимом шифрования **OfflineStorageMode** окончательное дешифрование данных производится до того, как **микросервер** отправит **resEncrMhStop**. Это также относится к любому окончательному дешифрованию данных, которые могут потребоваться для дешифрования в других типах сеансов.

9.7.2.5.12 Сообщение reqEncrMhQuit

C→H reqEncrMhQuit(ushort **mh,** uchar **reason) →**
C→H resEncrMhQuit(ushort **mh)**

- Это сообщение позволяет **микросерверу** информировать **хост ECI** о том, что операция повторного шифрования, связанная с **указателем медиаданных**, была прекращена.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для прекращенного сеанса шифрования
reason: uchar	Обоснование приведено в таблице 9.7.2.5.9-1.

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для прекращенного сеанса шифрования.

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** находился в запущенном состоянии, но в настоящий момент завершен.

Предварительные условия (отклик)

1) **Хост ECI** осведомлен о незапущенном состоянии шифрования сеанса.

Подробная семантика

- В случае если ошибка имеет квазипостоянный характер, **микросервер** также может отменить сеанс **указателя медиаданных** самостоятельно.
- В случае если **микросервер** может выдавать достоверные данные дешифрования перед завершением сеанса **повторного шифрования**, в сеансах с режимом шифрования **OfflineStorageMode** окончательные результаты дешифрования выдаются до того как **микросервер** отправит **resEncrMhQuit**. Это также относится к любому окончательному дешифрованию данных, которые могут потребоваться для дешифрования в других типах сеансов.

9.7.2.5.13 Сообщение reqEncrIpServer

H→C reqEncrIpServer(ushort **mh) →**
C→H resEncrIpServer(ushort **mh.** Addrinfo **addr)**

- Это сообщение позволяет **хосту ECI** направлять **микросерверу** запрос на предоставление IP-адреса **целевого объекта** для входящих IP-соединений от **микроклиентов**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования, в котором требуется IP-адрес для входящих сообщений или соединений

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования, в котором требуется IP-адрес для входящих сообщений или соединений
addr: Addrinfo	Протокол/адрес/порт IP для входящих сообщений или соединений **микроклиента**

Предварительные условия (запрос)

1) Сеанс указателя медиаданных открыт в режиме **OnlineIpMode**.

Предварительные условия (отклик)

1) **Хост ECI** осведомлен о незапущенном состоянии шифрования сеанса.

Подробная семантика

- Обмен данными по IP-протоколу между **микроклиентом** и **микросервером** характерен для **микросистем DRM**. Это включает выбор протокола и все условные обозначения для прекращения соединения или обмена в сеансе потоковой передачи контента.
- Сообщение может быть направлено по сеансу **указателя медиаданных**, в котором процесс повторного шифрования еще не начался.

Таблица 9.7.2.5.13-1 – Коды ошибок reqEncrIpServer

Название	Описание
ErrEncrIpNone	См. таблицу 9.7.2.5.19-1

9.7.2.5.14 Сообщение reqEncrMsgSend

C→H reqEncrMsgSend(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resEncrMsgSend(ushort **mh)**

- Это сообщение позволяет **микросерверу** направлять **хосту ECI** запрос на переадресацию сообщения **целевому микроклиенту** или **микроклиентам** (для групповых целевых объектов) относительно сеанса **указателя медиаданных**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования, сообщение для которого должно быть переадресовано **целевому микроклиенту**
length: uint	Длина поля **msg** в байтах
msg[]: byte	Сообщение для переадресации **микроклиенту**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования

Предварительные условия (запрос)

1) Сеанс указателя медиаданных открыт в режиме **OnlineMsgMode**.

Предварительные условия (отклик)

1) Сообщение переадресовано **микроклиенту**; **хост ECI** готов принять новое сообщение **reqEncrMsgSend**.

Подробная семантика

- **Хост ECI** должен быть способен обрабатывать и переадресовывать **микроклиенту** как минимум одно сообщение одновременно. Сообщения должны передаваться в установленном порядке. **Хост ECI** не обязан предоставлять специальную буферизацию более чем для одного одновременно выполняющегося запроса **reqEncrMsgSend**. При безопасном внедрении

микросервера должно использоваться **resEncrMsgSend** как подтверждение установления связи в процессе управления.

- Механизм переадресации **хоста ECI** должен обладать надежностью, достаточной для того, чтобы в стандартных приложениях не возникали сбои (потери сообщений или нарушение упорядоченности в 1 случае из 10 000). В приложениях, в которых важная информация для доступа к зашифрованному контенту может быть утрачена навсегда или просмотр важной информации может быть затруднен, рекомендуется принимать дополнительные меры предосторожности на уровне приложений.

9.7.2.5.15 Сообщение reqEncrMsgRecv

H→C reqEncrMsgRecv(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resEncrMsgRecv(ushort **mh)**

- Это сообщение разрешает **хосту ECI** передавать **микросерверу** сообщение от **целевого микроклиента**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования, для которого **микросервер** получает сообщение от **целевого микроклиента**
length: uint	Длина поля **msg** в байтах
msg: byte[]	Сообщение для получения **микросервером**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования, в котором требуется IP-адрес для входящих сообщений или соединений

Предварительные условия (запрос)

1) Сеанс указателя **медиаданных** открыт в режиме **OnlineMsgMode**.

Предварительные условия (отклик)

1) Сообщение обработано **микросервером**, и он готов к приему нового сообщения **reqEncrMsgRecv**.

Подробная семантика

- **Микросервер** должен обрабатывать как минимум одно сообщение за один раз. **Микросервер** не обязан обеспечивать специальную буферизацию более чем для одного одновременно выполняющегося запроса **reqEncrMsgSend**. В то же время он должен быть готов к обработке последующего сообщения, что связано с прочими требованиями к скорости отклика. При безопасном внедрении **хоста ECI** должно использоваться **resEncrMsgRecv** как подтверждение установления связи в процессе управления.
- Бесперебойная работа службы переадресации между **микроклиентом** и **микросервером** определена для **reqEncrMsgSend** в пункте 9.7.2.5.14.

9.7.2.5.16 Сообщение reqEncrTsData

C→H reqEncrTsData(ushort **mh,** TsSync **sync,** uint **length,** byte **msg[]) →**
C→H resEncrTsData(ushort **mh)**

- Это сообщение позволяет **микросерверу** предоставлять **хосту ECI** данные для переадресации **целевому микроклиенту указателя медианных** для дешифрования контента, включая информацию синхронизации для сообщений ECM.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования
sync: TsSync	Синхронизация информации, относящейся к ecmId, связанному с контентом. Подробная информация приведена в таблице 9.7.2.5.16-1
length: uint	Длина переадресуемого сообщения в байтах
msg: byte[]	Сообщение для переадресации **микроклиенту**

Таблица 9.7.2.5.16-1 – Определение TsSync typedef

```
typedef struct TsSync {
    uint   ecmId;
    uint   precTime;
} TsSync;
```

Определения полей

ecmId: uint	Идентификационный номер ECM, связанного с контентом, которому должно предшествовать сообщение, передающее данные для **микроклиента**
precTime: uint	Режим реального времени с интервалами 100 мс (максимальное значение 300 секунд), относящийся ко времени воспроизведения контента; данное сообщение должно передаваться до использования сообщения ECM с идентификатором ecmId в процессе декодирования контента

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования, в котором требуется IP-адрес для входящих сообщений или соединений

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** открыт в *режиме повторного шифрования* **OfflineStream** или в режиме **OfflineStorage**, использует *режим формата данных* **OfflineDataMode** и *режим синхронизации* **SyncTs**.

Предварительные условия (отклик)

1) **Хост ECI** готов получить следующее сообщение, содержащее данные.

Подробная семантика

- **Хост ECI** должен гарантировать, что **микроклиент** обеспечен данными в соответствии с требованиями к синхронизации, а также зашифрованным контентом.
- **Хост ECI** должен соответствующим образом буферизировать данные в сообщении (в качестве связанных с контентом данных) и отвечать на следующее в течение периода времени, предлагаемого в [b-ITU-T J Suppl. 7].
- **Микросервер** может формировать одно или несколько сообщений, содержащих данные, перед началом **сеанса повторного шифрования** при работе в режиме **OfflineStream**.
- **Микросервер** должен формировать не более одного сообщения, содержащего данные, в конце сеанса шифрования в режиме **OfflineStorage**. Сообщению, содержащему данные, может предшествовать сообщение ECM, с которым оно должно синхронизироваться (соответственно, режим "офлайн-хранения"). Как правило, это сообщение, содержащее данные, должно обрабатываться **микроклиентом** перед любым другим контентом и сообщениями ECM.

9.7.2.5.17 Сообщение reqEncrTsEcm

C→H reqEncrTsEcm(ushort **mh,** uint **ecmId,** uint **length,** byte **ecm[]) →**
C→H resEncrTsEcm(ushort **mh)**

- Это сообщение позволяет **микросерверу** предоставлять секцию ECM, запрошенную для дешифрования в следующем криптопериоде.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования
ecmId: uint	Идентификационный номер сообщения ECM, присваиваемого **микросервером** для синхронизации сообщений обмена данными
length: uint	Длина параметра **ecm** в байтах; ecm имеет формат отдельной секции
ecm: byte[]	Сообщение ECM, которое вводится в следующем криптопериоде

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** открыт в режиме *synchronization-mode* **SyncTs**.

Предварительные условия (отклик)

1) **Хост ECI** готов ввести следующее сообщение ECM.

Подробная семантика

- **Хост ECI** должен вводить сообщение ECM в транспортный поток в пределах определенного временного интервала после получения сообщения. Значения временного интервала предлагаются в [b-ITU-T J Suppl. 7]. Сообщение ECM должно повторяться в течение приемлемого временного интервала (как указано в [ISO/IEC 13818-1-1]. PID сообщения ECM является свободным PID и генерируется **хостом ECI**.
- **Хост ECI** может обновить всю информацию PMT в потоке, которая может отражать PID ECM, или, в противном случае, переадресовать информацию по PID ECM, что позволит **микроклиенту** позже восстановить информацию, необходимую для дешифрования.
- При изменении элемента контента и/или других общих изменениях в процессе шифрования **микросервер** может сформировать два последовательных, но отличающихся сообщения ECM для одного предстоящего криптопериода. В оставшийся период **хост ECI** должен ввести как минимум последнее из них. В режиме со смещением времени/хранением данных хост ECI вводит последнее сообщение ECM для всего криптопериода.

9.7.2.5.18 Сообщение reqEncrFileData

H→C reqEncrFileData(ushort **mh,** byte **syncKid[**MaxUuidLen], uint **datalength,** byte **data[])**
C→H resEncrFileData(ushort **mh)**

- Это сообщение разрешает **микросерверу** предоставить **хосту ECI** сообщение для передачи **целевому микроклиенту указателя медиаданных** для дешифрования, включая информацию для синхронизации, относящуюся к KeyID.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса шифрования
syncKid [MaxUuidLen]**:** byte	KeyId, который будет использоваться для шифрования следующего "фрагмента" файла, для которого требуются соответствующие данные, необходимые **микроклиенту** для дешифрования
datalength: uint	Длина блока данных в байтах
data[]**:** byte	Данные, предназначенные **микроклиенту** для дешифрования. Формат данных не определен в том случае, если режим формата данных – **OfflineDataMode**, и представляет формат блока PSSH, включаемого в блок ISOBMFF MOOV или MOOF, в том случае, если режим формата данных – **OfflineIsobmffMode**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** открыт в *режиме повторного шифрования* **OfflineStream** или в режиме **OfflineStorage** и *режиме синхронизации* **SyncFile**.

Предварительные условия (отклик)

1) **Хост ECI** готов получить следующее сообщение, содержащее данные.

Подробная семантика

- **Хост ECI** должен гарантировать, что каждый **целевой микроклиент** обеспечен данными в соответствии с требованиями к синхронизации наряду с зашифрованным контентом.
- Хост ECI должен создавать действительный файл ISOBMFF, включающий предоставленный блок PSSH, или, в противном случае, обеспечивать передачу данных вместе с содержимым файла **микроклиенту** и предоставление их **микроклиенту** в соответствии с требованиями по синхронизации.
- **Хост ECI** выполняет буферизацию данных в сообщении **reqEncrMsgRecv** в установленном порядке (в качестве данных, связанных с контентом). Требуемые значения времени **отклика** предлагаются в [b-ITU-T J Suppl. 7].
- **Микросервер** может формировать одно или несколько сообщений, содержащих данные, перед началом **сеанса повторного шифрования** при работе в режиме **OfflineStream**.
- **Микросервер** должен формировать не более одного сообщения, содержащего данные, в конце сеанса шифрования в режиме **OfflineStorage**. Как правило, это сообщение, содержащее данные, должно обрабатываться **микроклиентом** перед любым другим контентом.

9.7.2.5.19 Коды ошибок для API повторного шифрования

Таблица 9.7.2.5.19-1 – Коды ошибок для API повторного шифрования

Название	Значение	Описание
ErrEncrAuthInconclusive	1	Аутентификация выполнялась лишь частично и была не окончательной, однако никакой ошибки не произошло
ErrEncrAuthFail	–256	Определение статуса родительской аутентификации элемента контента выполнить не удалось, однако родительская аутентификация проведена корректно
ErrEncrUserMissing	–257	**Пользователь** не предоставляет **микросерверу** существенных данных для продолжения повторного шифрования контента
ErrEncrCardMissing	–258	**Смарт-карта**, требуемая для повторного шифрования, не может быть подключена (повторно подключена) и использована для повторного шифрования контента в течение приемлемого временного интервала
EnrEncrServiceMissing	–259	Служба, не относящаяся к **CPE** и поддерживающая **микросервер** в сеансе дешифрования, недоступна в течение приемлемого временного интервала
ErrEncrResourceMissing	–260	Неизвестный ресурс в составе оборудования **CPE**, необходимый для обработки и/или повторного шифрования контента, недоступен
ErrEncrMmiMissing	–261	Доступ **микросервера** к интерфейсу MMI требуется, но отсутствует
ErrEncrClientAuthError	–262	**Микросерверу** не удалось выполнить аутентификацию **целевого микроклиента**
ErrEncrIpNone	–263	**Микросервер** не может предоставить IP-адрес для связи с **микроклиентом**

9.7.2.6 API дешифрования микроклиента

9.7.2.6.1 Общие сведения

API дешифрования **микроклиента** позволяет **микроклиенту** дешифровать контент, полученный от **микросервера**.

На этапе обнаружения **микроклиенту** разрешено объявлять целевые объекты дешифрования, которым он может оказывать услуги дешифрования и предоставлять идентификационные данные, по которым **микросервер** может создавать аутентифицированное соединение с **микроклиентом** как с целевым объектом.

Микроклиент должен поддерживать режимы дешифрования, которые охватывают режимы шифрования, предоставляемые его дополняющим **микросервером**. **Микроклиент** может выполнять услуги дешифрования, используя один из общепринятых режимов, основанных на общем API дешифрования.

Дополнительные сообщения прямой и обратной передачи данных для необходимого дешифрования между **микросервером** и **микроклиентом** в различных режимах являются частью данного API.

Сообщения для API дешифрования **микроклиента** перечислены в таблице 9.7.2.6.1-1.

Таблица 9.7.2.6.1-1 – Сообщения API дешифрования

Сообщение	Тип	Направление	Маркер	Описание
setDcrModes	set	C→H	0x0	**Микроклиент** информирует **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации)
reqDcrTargets	A	H→C	0x1	**Хост ECI** направляет **микроклиенту** запрос на предоставление целевых объектов шифрования, которым он может оказать услуги дешифрования
reqDcrTargetCred	A	H→C	0x2	**Хост ECI** направляет **клиенту ECI** запрос на предоставление данных инициализации для соединения **микросервера**, обычно используемого для аутентификации целевого объекта
reqDcrIpServer	A	C→H	0xA	**Микроклиент** направляет **хосту ECI** запрос на предоставление IP-адреса **микросервера** для дополнительной связи, относящейся к сеансу **указателя медиаданных**
reqDcrMsgSend	A	C→H	0xB	**Микроклиент** направляет **хосту ECI** запрос на отправку сообщения **микросерверу** относительно сеанса **указателя медиаданных**
reqDcrMsgRecv	A	H→C	0xC	**Хост ECI** передает **микроклиенту** сообщение от **микросервера** относительно сеанса **указателя медиаданных**
reqDcrTsData	A	C→H	0xD	**Микросервер** предоставляет **хосту ECI** данные для переадресации **целевому микроклиенту указателя медианных** для дешифрования, включая информацию синхронизации для сообщений ECM
reqDecrFileData	A	C→H	0xF0	**Микросервер** предоставляет **хосту ECI** сообщение для переадресации **целевому микроклиенту указателя медианных** для дешифрования, включая информацию синхронизации для KeyID

9.7.2.6.2 Сообщение setDcrModes

C→H setDcrModes(EciEncrModes **modes)**

- Это сообщение позволяет **микроклиенту** информировать **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации.

Определение параметра запроса

modes: EciEncrModes	Режимы дешифрования, поддерживаемые **микроклиентом**. Тип EciEncrModes определен в таблице 9.7.1.5.2-1

9.7.2.6.3 Сообщение reqDcrTargets

H→C reqDcrTargets() →
C→H resDcrTargets(EncrTarget **target[])**

- Это сообщение позволяет **хосту ECI** направлять **микроклиенту** запрос на предоставление целевых объектов шифрования, для которых он может выполнять дешифрование.

Определение параметра отклика

target[]: EncrTarget	Список целевых объектов шифрования, которые может аутентифицировать **микросервер**. Определение типа TargetClient приведено в таблице 9.7.2.5.2-1

Подробная семантика

- **Хост ECI** может соответствовать **потенциальным целевым микроклиентам** на основе **целевого объекта**. Определение расположения потенциальных **микроклиентов** производится на усмотрение приложения и/или **хоста ECI**.

9.7.2.6.4 Сообщение reqDcrTargetCred

H→C reqDcrTargetsCred(EncrTarget **target) →**
C→H reqDcrTargetsCred(uint **credLen**, byte **cred[]**)

- Это сообщение позволяет **хосту ECI** направлять **микроклиенту** запрос на предоставление идентификационных данных для шифрования **микросервером**.

Определение параметра запроса

target: EncrTarget[]	**Целевой объект** шифрования, которому **микроклиент** должен предоставлять фактические идентификационные данные для шифрования контента **микросервером**

Определение параметра отклика

credLen: uint	Длина параметра cred в байтах.
cred[]: byte	Идентификационные данные, закодированные в специальном формате для **микросервера**, который шифрует контент, подлежащий дешифрованию **микроклиентом**

Подробная семантика

- Это сообщение позволяет **хосту ECI** направлять **микроклиенту** запрос на предоставление идентификационных данных, соответствующих **целевому параметру**, таким образом, чтобы **микросервер**, распознающий **целевой объект**, мог шифровать контент для **микроклиента**.

9.7.2.6.5 Сообщение reqDcrIpServer

C→H reqDcrIpServer(ushort **mh) →**
C→H resDcrIpServer(ushort **mh,** Addrinfo **addr)**

- Это сообщение позволяет **микроклиенту** направлять **хосту ECI** запрос на предоставление IP-адреса **микросервера** для дополнительной связи, относящейся к сеансу **указателя медиаданных**. Соответствующие коды ошибок определены в таблице 9.7.2.6.5-1.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса дешифрования, для которого запрашиваются сообщения отправки/приема IP-адреса **микросервера**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса дешифрования, для которого предоставляются сообщения отправки/приема IP-адресов **микросервера**
addr: Addrinfo	Протокол/адрес/порт IP для **микросервера** для данного указателя **медиаданных**

Предварительные условия (запрос)

1) Сеанс указателя **медиаданных** открыт в режиме **OnlineIpMode**.

Предварительные условия (отклик)

1) **Хост ECI** осведомлен о незапущенном состоянии шифрования сеанса.

Подробная семантика

- Обмен данными по IP-протоколу между **микроклиентом** и **микросервером** характерен для **микросистемы DRM**. Это включает выбор протокола и все условные обозначения для прекращения соединения или обмена в сеансе потоковой передачи контента.
- Сообщение может быть направлено по сеансу **указателя медиаданных**, в котором процесс повторного шифрования еще не начался.

Таблица 9.7.2.6.5-1 – Коды ошибок reqDcrIpServer

Название	Описание
ErrDcrIpNone	См. таблицу 9.7.2.6.10-1

9.7.2.6.6 Сообщение reqDcrMsgSend

C→H reqDcrMsgSend(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resDcrMsgSend(ushort **mh)**

- Это сообщение позволяет **микроклиенту** направлять **хосту ECI** запрос на переадресацию сообщения **целевому микросерверу** относительно сеанса **указателя медиаданных**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса дешифрования, сообщение для которого должно быть переадресовано **микросерверу**
length: uint	Длина поля **msg** в байтах
msg[]: byte	Сообщение для переадресации **микросерверу**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса дешифрования

Предварительные условия (запрос)

1) Сеанс указателя **медиаданных** открыт в режиме **OnlineMsgMode**.

Предварительные условия (отклик)

1) Сообщение переадресовано **микросерверу**; **хост ECI** готов принять новое сообщение **reqDcrMsgSend**.

Подробная семантика

- **Хост ECI** должен быть способен обрабатывать и переадресовывать **микросерверу** как минимум одно сообщение одновременно. Сообщения должны передаваться в установленном порядке. **Хост ECI** не обязан предоставлять специальную буферизацию более чем для одного одновременно выполняющегося запроса **reqDcrMsgSend**. При безопасном внедрении **микроклиента** должно использоваться **resDcrMsgSend** как подтверждение установления связи в процессе управления.
- Бесперебойная работа службы переадресации между **микросервером** и **микроклиентом** определена для **reqEncrMsgSend** в пункте 9.7.2.5.14.

9.7.2.6.7 Сообщение reqDcrMsgRecv

H→C reqDcrMsgRecv(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resDcrMsgRecv(ushort **mh)**

- Это сообщение разрешает **хосту ECI** передавать **микроклиенту** сообщение от **целевого микросервера**.

Определение параметра запроса

mh: ushort	Указатель медиаданных для сеанса дешифрования, для которого **микроклиент** получает сообщение от **микросервера**
length: uint	Длина поля **msg** в байтах
msg[]: byte	Сообщение для получения от **микросервера**

Определение параметра отклика

mh: ushort	**Указатель медиаданных для сеанса дешифрования**

Предварительные условия (запрос)

1) Сеанс указателя **медиаданных** открыт в режиме **OnlineMsgMode**.

Предварительные условия (отклик)

1) Сообщение обработано **микроклиентом**, и он готов к приему нового сообщения **reqDcrMsgRecv**.

Подробная семантика

- **Микроклиент** должен обрабатывать как минимум одно сообщение за один раз. **Микроклиент** не обязан обеспечивать специальную буферизацию более чем для одного одновременно выполняющегося запроса **reqDcrMsgSend**. В то же время он должен быть готов к обработке последующего сообщения, что связано с прочими требованиями к скорости отклика. При безопасном внедрении **хоста ECI** должно использоваться **resDcrMsgRecv** как подтверждение установления связи в процессе управления.

- Бесперебойная работа службы переадресации между **микроклиентом** и **микросервером** определена для **reqEncrMsgSend** в пункте 9.7.2.5.14.

9.7.6.2.8 Сообщение reqDcrTsData

H→C reqDcrTsData(ushort **mh,** uint **length,** byte **msg[]) →**
C→H resDcrTsData(ushort **mh)**

- Это сообщение разрешает **хосту ECI** передавать **микроклиенту** данные, необходимые **указателю медиаданных** для дешифрования контента в (ближайшем) будущем.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса дешифрования
length: uint	Длина переадресуемого сообщения в байтах
msg[]: byte	Сообщение для переадресации **микроклиенту**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса дешифрования

Предварительные условия (запрос)

1) Сеанс **указателя медиаданных** открыт в *режиме повторного шифрования* **OfflineStream** или в режиме **OfflineStorage**, использует *режим формата данных* **OfflineDataMode** и *режим синхронизации* **SyncTs**.

Предварительные условия (отклик)

1) **Хост ECI** готов получить следующее сообщение, содержащее данные.

Подробная семантика

- **Хост ECI** должен гарантировать, что **микроклиент** обеспечен данными в соответствии с требованиями к синхронизации, предъявляемыми **микросервером**, а также зашифрованным контентом, подлежащим дешифрованию.
- **Микроклиент** должен формировать не более одного сообщения, содержащего данные, в начале сеанса дешифрования в режиме **OfflineStorage** (соответственно, режим "офлайн-хранения").

9.7.2.6.9 Сообщение reqDcrFileData

H→C reqDcrFileData(ushort **mh,** uint **datalength,** byte **data[])**
C→H resDcrFileData(ushort **mh)**

- Это сообщение разрешает **хосту ECI** передавать **микроклиенту** от **целевого микросервера** сообщение, необходимое **указателю медиаданных** для дешифрования контента.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** для сеанса дешифрования
datalength: uint	Длина блока данных в байтах
data[]: byte	Данные, предназначенные **микроклиенту** для дешифрования. Формат данных не определен в том случае, если режим формата данных – **OfflineDataMode**, и представляет формат блока PSSH, включаемого в блок ISOBMFF MOOV или MOOF, в том случае, если режим формата данных – **OfflineIsobmffMode**

Определение параметра отклика

mh: ushort	**Указатель медиаданных** для сеанса шифрования

Предварительные условия (запрос)

1) Сеанс указателя **медиаданных** открыт в режиме повторного шифрования OfflineStream или в режиме OfflineStorage и режиме синхронизации SyncFile.

Предварительные условия (отклик)

1) Микроклиент готов получить следующее сообщение, содержащее данные.

Подробная семантика

- **Хост ECI** должен гарантировать, что **микроклиент** обеспечен данными в соответствии с требованиями к синхронизации, а также зашифрованным контентом.
- Хост **ECI** может извлечь блок PSSH из действительного файла ISOBMFF и передать его **микроклиенту** в соответствии с требованиями к синхронизации для декодирования файлов ISOBMFF.
- **Хост ECI** должен формировать не более одного сообщения, содержащего данные, в конце сеанса шифрования в режиме **OfflineStorage**. Как правило, это сообщение, содержащее данные, должно обрабатываться **микроклиентом** перед любым другим контентом.

9.7.2.6.10 Коды ошибок для API дешифрования микроклиента

Коды ошибок для API дешифрования микроклиента перечислены в таблице 9.7.2.6.10-1.

Таблица 9.7.2.6.10-1 – Коды ошибок, связанные с интерфейсом дешифрования микроклиента

Название	Значение	Описание
ErrDcrIpNone	–256	У **хоста ECI** нет адреса/порта IP для связи с **микросервером**

9.8 Интерфейсы API для ресурсов, связанных со свойствами контента

9.8.1 Список интерфейсов API, приведенных в пункте 9.8

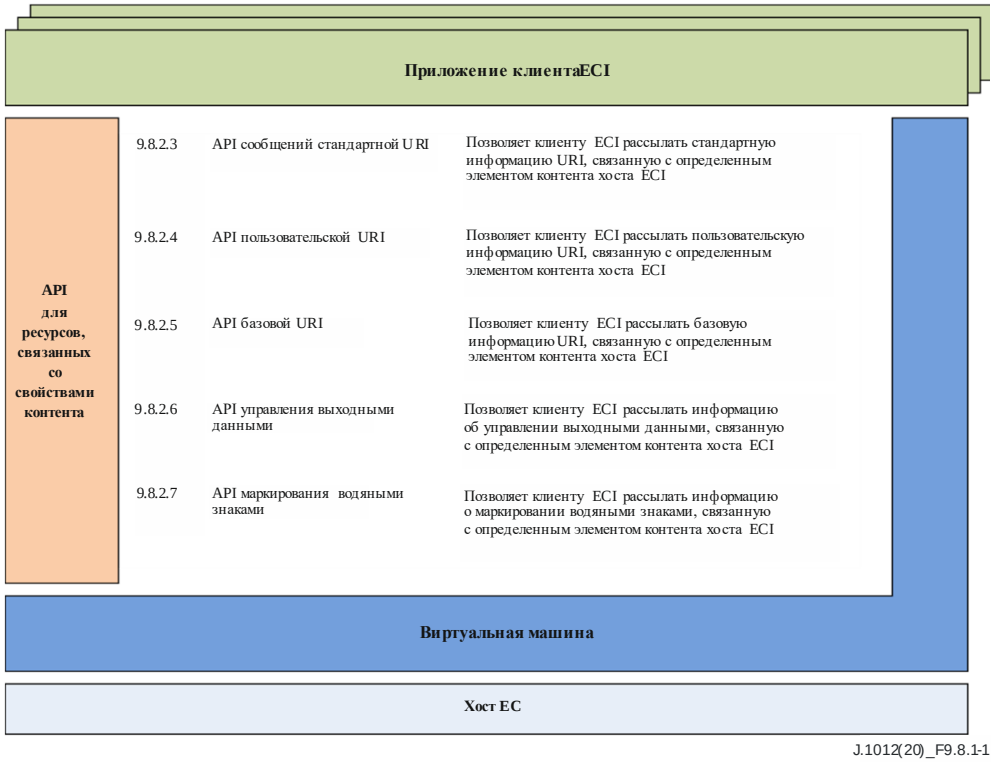


Рисунок 9.8.1-1 – Блок-схема интерфейсов API, определенных в пункте 9.8

В таблице 9.8.1-1 перечислены интерфейсы API, рассмотренные в пункте 9.8, а рисунок 9.8.1-1 иллюстрирует расположение интерфейсов API, определенных в пункте 9.8, а также **архитектуру ECI**.

Таблица 9.8.1-1 – Интерфейсы API для ресурсов, связанных с защитой контента

Пункт	Наименование API	Описание
9.8.2.3	API сообщения стандартной URI	Позволяет **клиенту ECI** передавать **хосту ECI** и обратно стандартную информацию URI, относящуюся к определенному элементу контента
9.8.2.4	API пользовательской URI	Позволяет **клиенту ECI** передавать **хосту ECI** и обратно **пользовательскую** информацию URI, относящуюся к определенному элементу контента
9.8.2.5	API базовой URI	Позволяет **клиенту ECI** передавать **хосту ECI** и обратно базовую информацию URI, относящуюся к определенному элементу контента
9.8.2.6	API управления выходными данными	Позволяет **клиенту ECI** передавать **хосту ECI** и обратно информацию по управлению выходными данными, относящуюся к определенному элементу контента
9.8.2.7	API маркирования водяными знаками	Позволяет **клиенту ECI** передавать **хосту ECI** и обратно информацию о нанесении водяных знаков, относящуюся к определенному элементу контента
9.8.2.8	API родительского контроля	Позволяет **клиенту ECI** передавать **хосту ECI** информацию об обязательствах по родительскому контролю, относящуюся к определенному элементу контента
9.8.2.9	API синхронизации свойств контента	Разрешает синхронизацию различных изменений свойств контента
9.8.2.10	API родительской аутентификации	Позволяет **клиенту ECI** делегировать родительскую аутентификацию стандартной функции в **хосте ECI**
9.8.2.11	API делегирования родительской аутентификации	Позволяет **клиенту ECI** отменять запрос о делегировании родительской аутентификации
9.8.2.12	API управления защитой	Позволяет **клиенту ECI** обеспечивать управление системами защиты выходных данных с учетом специфики **оператора платформы**

9.8.2 API для доступа к ресурсу прав на использование и родительского контроля

9.8.2.1 Введение

В настоящем пункте, посвященном интерфейсам API **клиента/хоста ECI**, **клиенту ECI** разрешено безопасным способом устанавливать права и условия, применяемые к дешифрованному контенту.

API прав и условий определяет следующие аспекты.

- Стандартная информация URI (информация о правах пользователя): генерируется **клиентом ECI** и используется **хостом ECI** для управления приложениями контента, связанного с отраслевыми стандартными выходными данными и приложениями.
- Базовая информация URI: генерируется **клиентом ECI** и используется **усовершенствованной системой безопасности** и аппаратной подсистемой **хоста ECI** для установки базовых прав на использование контента. Это позволяет **клиенту ECI** использовать устойчивую аппаратную защиту для свойств базовых прав, которые должны присутствовать в контенте.
- Управление выходными данными: позволяет **клиенту ECI** выборочно блокировать выходные данные, которые могут быть активными при условиях, установленных в URI, однако при этом считаются некорректными с правовой точки зрения.
- Управление маркированием водяными знаками с использованием **хоста ECI**: позволяет **клиенту ECI** маркировать выходной контент отметками, определенными **клиентом ECI**, используя систему маркирования водяными знаками, встроенную в оборудование **CPE**.
- Условия родительского контроля позволяют **клиенту ECI** переадресовывать требование об аутентификации родителей для предоставления доступа к контенту системе защиты, в которую экспортируется контент.
- Синхронизация свойств контента допускает одновременное изменение нескольких свойств контента, которые должны быть идентифицированы как таковые.
- Функция родительской аутентификации может выполняться **клиентом ECI** или может быть делегирована централизованной стандартной функции **хоста ECI**. **Хост ECI**, в свою очередь, может выбирать конкретного **клиента ECI** для выполнения родительской аутентификации от своего имени. Параметры делегирования позволяют выполнять одну родительскую аутентификацию с несколькими **клиентами ECI** и **хостом ECI**.

Применение новых свойств прав безопасным образом связано с применением нового слова управления для дескремблирования контента. Это гарантирует, что права применяются к тому контенту, с которым они связаны.

Интерфейсы API свойств контента используют сообщение установки (*set*) и получения (*get*). Сообщение *set* используется **клиентами ECI**, выполняющими дешифрование контента, для передачи информации о свойствах контента, связанных со следующим рассчитанным словом управления. Функция *get* используется **микросерверами**, выполняющими повторное шифрование контента, для получения свойств входящего контента в целях формирования надлежащей аутентификации и передачи данных для свойств сигнализации повторно зашифрованного контента.

Версия API, передаваемая как часть API обнаружения, успешно согласуется с версией используемых свойств контента.

Контекст **указателя медиаданных хоста ECI** должен поддерживать как минимум два значения для различных секций каждого из свойств контента. Что касается дешифрования на основе файлов, должны поддерживаться по меньшей мере две секции контента, каждая из которых декодируется отдельным KeyID для каждого из свойств контента. В таблице 9.8.2.1-1 перечислены функции API. Функции API прав сгруппированы в отдельные API, что позволяет управлять версиями независимо друг от друга.

Таблица 9.8.2.1-1 – Список сообщений интерфейса API, касающихся прав на использование и родительского контроля

API	Сообщение	Тип	Направление	Маркер	Описание
ApiStdUri	setDcrStdUri	set	C→H	0x0	Установка стандартной URI для контента, подлежащего дескремблированию
ApiStdUri	getEncrStdUri	get	C→H	0x1	Получение стандартной URI для контента, подлежащего повторному шифрованию
ApiCustUri	setDcrCustUri	set	C→H	0x0	Установка пользовательской URI для контента, подлежащего дескремблированию
ApiCustUri	getEncrCustUri	get	C→H	0x1	Получение пользовательской URI для контента, подлежащего повторному шифрованию
ApiBasicUri	setDcrBasicUri	set	C→H	0x0	Установка базовой URI для контента, подлежащего дескремблированию
ApiBasicUri	getEncrBasicUri	get	C→H	0x1	Получение базовой URI для контента, подлежащего повторному шифрованию
ApiOC	setDcrOutputCtl	set	C→H	0x0	Установка ограничений по управлению выходными данными для контента, подлежащего дескремблированию
ApOC	getEncrOutputCtrl	get	C→H	0x1	Получение ограничений по управлению выходными данными для контента, подлежащего повторному шифрованию
ApiDcrMark	getDcrMarkSyst	get	H→C	0x0	Получение поддерживаемых систем маркирования.
ApiDcrMark	setDcrMarkMeta	set	C→H	0x1	Установка контрольного значения системы маркирования
ApiDcrMark	getDcrMarkMeta	get	H→C	0x2	Считывание свойства системы маркирования
ApiDcrMark	setDcrMarkBasic	set	C→H	0x3	Установка базовых данных для маркирования контента, подлежащего дескремблированию
ApiDcrMark	setDcrMarkExt	set	C→H	0x4	Установка расширенных полезных данных маркирования для контента, подлежащего дескремблированию
ApiPar	setDcrParCtl	set	C→H	0x0	Установка условий родительского контроля для контента, подлежащего дескремблированию
ApiPar	getEncrParCtrl	get	C→H	0x1	Получение условий родительского контроля для контента, подлежащего дескремблированию
ApiCpSync	setCpSync	set	C→H	0x0	**Клиент ECI** посылает информацию о том, что текущий набор свойств контента согласован и может применяться к контенту, подлежащему дескремблированию с использованием слова управления в будущем
ApiCpSync	reqCpChange	req	H→C	0x1	**Хост ECI** посылает информацию о том, что предстоит изменение свойств контента, подлежащего повторному шифрованию

Таблица 9.8.2.1-1 – Список сообщений интерфейса API, касающихся прав на использование и родительского контроля

ApiParAuth	reqParAuthChk	req	C→H	0x0	Запрос **хосту ECI** на проведение родительской аутентификации от имени **клиента ECI**
ApiParAuth	reqParAuthChkCan	req	C→H	0x1	Отменяет предыдущий запрос родительской аутентификации, направленный хосту
ApiParAuth	reqParAuthCid	req	H→C	0x2	Запрашивает PIN-код для родительской авторизации для (будущего) элемента контента, подлежащего декодированию. Таким образом может быть запущен диалог родительской аутентификации
ApiParAuthDel	reqParAuthDel	req	H→C	0x0	**Хост ECI** делегирует родительскую аутентификацию **клиенту ECI**
ApiParAuthDel	reqParAuthDelCan	req	H→C	0x1	**Хост ECI** отменяет предыдущий запрос родительской аутентификации, направленный **клиенту ECI**
ApiProtCtrl	getProtSystCtrl	get	C->H	0x0	**Клиент ECI** получает от **хоста ECI** список систем управления выходными данными и поддержке ими SRM (системных сообщений о возобновлении) и услуг блокировки устройств по идентификатору
ApiProtCtrl	reqSrmMsg	req	C->H	0x1	**Клиент ECI** передает SRM системе защиты выходных данных
ApProtCtrl	reqInfoDevId	req	H->C	0x2	**Хост ECI** передает идентификатор устройства, для которого система защиты выходных данных предоставляет защищенный контент в сеансе дешифрования
ApiProtCtrl	reqBlockDevId	req	C->H	0x3	**Клиент ECI** передает идентификатор устройства, для которого система защиты выходных данных не должна предоставлять контент в сеансе дешифрования
ApProtCtrl	setBlockProtSyst	set	C->H	0x4	**Клиент ECI** сообщает, что система защиты считается недостаточной для защиты контента в сеансе дешифрования

9.8.2.2 Аспекты безопасности и синхронизация

Спецификация **ECI** позволяет выполнять аутентификацию вышеуказанной информации о свойствах контента **хостом ECI** в целях предотвращения несанкционированных операций с этой информацией. Данный механизм также гарантирует, что надлежащие параметры прав применяются к контенту, к которому они относятся. Определено в [ITU-T J.1014].

Что касается информации о свойствах контента, **хост ECI** может упростить аутентификацию информации о правах от имени **клиента ECI**, используя ключи в блоке усовершенствованной системы безопасности, что позволяет обеспечить наивысший уровень целостности аутентификации. Порядок использования служб AS **хоста ECI** для этих целей **клиенты ECI** определяют самостоятельно. Это также определено в [ITU-T J.1014].

Если особенности контента требуют обеспечения конкретных защитных свойств в рамках защиты выходных данных, но **хост ECI** не может обеспечить таких свойств (или их разновидностей, предоставляющих более надежную защиту), **хост ECI** не выводит контент и уведомляет **пользователя** соответствующим сообщением. Более детальные требования устанавливаются режимом обеспечения соответствия в **экосистеме ECI**.

9.8.2.3 API сообщений стандартной URI

9.8.2.3.1 Сообщение setDcrStdUri

C→H **setDcrStdUri(**ushort **mh,** byte **keyId**[MaxUuidLen], StdUri **stdUri**)

- Это сообщение задает значение стандартной URI, связанной с **keyId**, равным **uri**.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения следующего слова управления
stdUri: StdUri	Стандартная URI для контента определена в таблице 9.8.2.3.1-1. Семантика полей соответствует полям, указанным в [ETSI TS 103 205] и [b-CI Plus]

Таблица 9.8.2.3.1-1 – Спецификация типа стандартной URI

```
typedef struct StdUri {

    uint MajorVersion: 4;
    uint tmc: 1;/* trick_mode_control_info in [CI+ v1.4] */
    unit reserved1: 3;
    uint aps: 2;      /* aps_copy_control_info in [CI+ v1.4] */
    uint emi: 2;      /* emi_copy_control_info in [CI+ v1.4] */
    uint ict: 1;      /* ict_copy_control_info in [CI+ v1.4] */
    uint rct: 1;      /* rct_copy_control_info in [CI+ v1.4] */
    uint reserved2: 1;    /* reserved bit */
    uint dot: 1;      /* dot_copy_control_info in [CI+ v1.4] */
    uint rl: 8;       /* rl_copy_control_info in [CI+ v1.4] */

} StdUri;

Применяются следующие правила (выражения с полем должны оцениваться как "True"
истинными) в соответствии с [CI+v1.4]
emi == 0b00 || rct == 0b0
emi == 0b11 || (dot == 0b0 && rl == 0x00)
emi == 0b01 || tmc == 0b0

Значение поля protocol_version 0x03 указано для определения выше; другие значения
зарезервированы для использования в будущем.
```

Семантика поля StdUri

MajorVersion: uint: 4	Основная версия данной стандартной URI. **Клиенты ECI** должны задавать значение поля MajorVersion равным 0b0000. **Хосты ECI** внедряют все версии согласно их уровню соответствия для данного поля и интерпретируют все существенные значения как неиспользуемую URI; таким образом, права на использование не применяются
reserved1: unit: 3	Биты зарезервированы. Задается равным 0b000 **клиентом ECI** и игнорируется **хостами ECI**, соответствующими данной версии stdUri
reserved2: unit:1	Бит зарезервирован. Задается равным 0b0 **клиентом ECI** и игнорируется **хостами ECI**, соответствующими данной версии stdUri
Другие поля	Семантика указана для обозначенных полей CI Plus v1.4 URI [ETSI TS 103 205] в приведенном выше определении структуры

Подробная семантика

- В режиме дескремблирования транспортного потока URI применяется к контенту, подлежащему декодированию, с ключами, применяемыми к следующему ключу дешифрования. Подробные сведения о вычислении ключа дешифрования приведены в пункте 8.2.4.7 [ITU-T J.1014].
- **Клиент ECI** работает в режиме дешифрования.

9.8.2.3.2 Сообщение getEncrStdUri

C→H StdUri getEncrStdUri(ushort **mh,** byte **keyId**[MaxUuidLen])

- Это сообщение задает стандартную URI для будущего контента.

Определение свойства

- Стандартная URI определена в таблице 9.8.2.3.1-1.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId: byte[MaxUuidLen]	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения следующего слова управления

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.

9.8.2.4 API пользовательской URI

9.8.2.4.1 Сообщение setDcrCustUri

C→H setDcrCustUri(ushort **mh,** byte **keyId**[MaxUuidLen], unit **custUriLen**, byte ***custUri)**

- Это сообщение задает значение пользовательской URI, связанной с **keyId**, равным **uri**.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId: byte[MaxUuidLen]t	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения следующего слова управления
custUriLen: unit	Длина поля пользовательской URI в байтах
custUri: byte *	Пользовательская URI для контента определена в таблице 9.8.2.4.1-1. Байты 0 и 1 выступают в роли msB и lsB формата пользовательской URI. Все значения байтов 0 и 1 зарезервированы, за исключением 0x80, 0x00, что должно указывать на специальное значение следующих байтов для приложения

Таблица 9.8.2.4.1-1 – Спецификация типа пользовательской URI

Название	Значение, байт 0, 1	Описание
CustUriPrivate	0x80, 0x00	Значение байтов, следующих за байтом 1, частное. Правильная интерпретация остальных полей определяется посредством другого соединения между **клиентом ECI** и **микросервером** или системой защиты
RFU	Прочее	Зарезервировано для использования в будущем

Подробная семантика

- В режиме дескремблирования транспортного потока URI применяется к контенту, подлежащему декодированию, с ключами, применяемыми к следующему ключу дешифрования. Подробные сведения о вычислении ключа дешифрования приведены в пункте 8.2.4.7 [ITU-T J.1014].
- Для одного слова управления может быть установлено не более четырех отдельных пользовательских URI.\
- **Клиент ECI** работает в режиме дешифрования.

9.8.2.4.2 Сообщение getEncrCustUri

C→H custUri getEncrCustUri(ushort **mh,** byte **keyId**[MaxUuidLen], unit **custUriMaxLen)**

- Это сообщение получает пользовательскую URI для будущего контента.

Определение свойства

- Пользовательская URI определена в таблице 9.9.1-1.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId: byte[MaxUuidLen]	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
custUriMaxLen: uint	Максимальная длина результата конечной пользовательской URI (в байтах); весь дополнительный контент отсекается

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.

9.8.2.5 API базовой URI

9.8.2.5.1 Сообщение setDcrBasicUri

C→H **setDcrBasicUri(**ushort **mh,** byte **keyId**[MaxUuidLen], BasicUri **basicUri**)

- Это сообщение задает значение базового URI, связанного с **keyId**, равным **basicUri**. Базовый URI предоставляет упрощенную, но высокоустойчивую схему управления правами для дешифрованного контента.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
basicUri: BasicUri	Базовая URI для контента определена в таблице 9.8.2.5.1-1. Семантика полей соответствует полям, указанным в [ETSI TS 103 205]

Таблица 9.8.2.5.1-1 – Спецификация типа базовой URI

```
typdef byte   BasicUri;
```

Название	Биты	Описание
BasicUriVersion	7	Основная версия данной базовой URI. Если **хост ECI** не реализовал версию, **хост ECI** не должен разрешать дешифрование и использование контента. Значение 0b0 определяет версию 0. Все другие значения зарезервированы и не допускаются
BasicUriV0_0Ext	2..6	Зарезервировано для использования в будущем, не используется в v0.0. Единственное значение, определенное для данного поля, равно 0b00000. Другие значения не допускаются. **Хосты ECI**, внедряющие только базовую URI версии v0.0, игнорируют значения данного поля: то есть оно может использоваться в будущем для обратно совместимых расширений версии v0.0, например, в форме ослаблений управления правами версии v0.0
BasicUriV0_0	0,1	Базовая URI, версия 0.0. Значения и параметры данного поля определены в таблице 9.8.2.5.1-2

Таблица 9.8.2.5.1-2 – Определение базовой URI, V0.0

Название	Значение	Описание
NoBasicProtection	0b00	Управление правами не осуществляется через базовую URI
RedistributionProtected	0b01	Шифрование включено, предотвращение повторного воспроизведения отключено
ViewOnly	0b10	Шифрование включено, предотвращение повторного воспроизведения включено
ViewOnlyStrict	0b11	Шифрование включено, предотвращение повторного воспроизведения включено, выходные данные ограничиваются специально отобранными (защищенными) данными

Подробная семантика

- В режиме дескремблирования транспортного потока URI применяется к контенту, подлежащему декодированию, с ключами, применяемыми к следующему ключу дешифрования. Подробные сведения о вычислении ключа дешифрования приведены в пункте 8.2.4.7 [ITU-T J.1014].
- Базовый URI позволяет **клиенту ECI** управлять реализацией прав на наивысшем уровне устойчивости, поддерживаемом **хостом ECI**. **Хост ECI** управляет двумя механизмами защиты: шифрованием, гарантирующим, что контент всегда скремблируется на всех выходных интерфейсах или накопителях, и предотвращением повторного воспроизведения, гарантирующим, что зашифрованный контент может быть дескремблирован только при прямом соединении (то есть не может храниться). Подробная информация приведена в [ITU-T J.1015].
- **Клиент ECI** работает в режиме дешифрования.

9.8.2.5.2 Сообщение getEncrBasicUri

C→H **BasicUri getEncrBasicUri(**ushort **mh,** byte **keyId**[MaxUuidLen])

- Это сообщение получает базовый URI для будущего контента.

Определение свойства

- Базовый URI определен в таблице 9.8.2.5.1-1.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.

9.8.2.6 Управление выходными данными

9.8.2.6.1 Сообщение setDcrOutputCtl

C→H **setDcrOutputCtl(**ushort **mh,** byte **keyId**[MaxUuidLen], ushort **ocVector**)

- Устанавливает в соответствии с **ocVector** параметры управления выходными данными, связанными с **keyId**.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
ocVector: ushort	Вектор управления для стандартных выходных данных определен в таблице 9.8.2.6.1-1

Таблица 9.8.2.6.1-1 – Спецификация вектора управления выходными данными

Название	Биты	Описание
MajorVersion	15	Версия параметра ocVector. Для версии 1 определено значение 0b0. Все остальные значения зарезервированы и не допускаются. Если **хост ECI**, использующий **основную версию** 1, получает значение, отличное от 0xb0, то вывод данных не разрешается
OcAnyOther	14	Любые другие выходные данные **хоста ECI**, не охваченные ни одним из квалификационных критериев выходных данных, перечисленных ниже, не охватывает. Если значение равно 0b0, вывод данных через указанные выходные интерфейсы разрешен; если значение равно 0b1, вывод данных не разрешен. ***Значение этого бита изменяет шифрование полей, указанных ниже.*** Если значение равно 0b0, ограничения, накладываемые на выходные данные, должны быть идентичны указанным ниже. Если значение равно 0b1, кодировка должна быть поразрядно инвертируемой. То есть если OcAnyOther==0b1 и OcIP==0b1, вывод данных через IP-соединение разрешен. См. примечание 2
OcIP	0	Вывод данных через IP-соединение разрешен, если значение равно 0b0, и не разрешен, если значение равно 0b1
OcUSB	1	Вывод любых данных через USB-соединение разрешен, если значение равно 0b0, и не разрешен, если значение равно 0b1. Предварительные условия: дешифрованный контент не защищен какой-либо системой защиты выходных данных, совместимой с **ECI**, и/или **микросистемой DRM ECI**, управляемой **клиентом ECI**, выполняющим дешифрование
OcDtcpIp	2	Вывод данных через защищенное соединение DTCP-IP разрешен, если значение равно 0b0, и не разрешен, если значение равно 0b1
OcHdcp	3,4	Все выходные данные, защищенные HDCP При OcAnyOther, равном 0b0: • значение 0b00: выходные данные, защищенные HDCP, разрешены; • значение 0b01: если версия HDCP ниже 2.2, вывод данных не разрешен, если версия HDCP равна 2.2 или выше, вывод данных разрешен; • значение 0b10: зарезервировано; значение не допускается. **Хосты ECI** должны интерпретировать это значение как равное 0b11; • значение 0b11: вывод данных, защищенный HDCP, не разрешен. При OcAnyOther, равном 0b1: • значение 0b00: вывод данных, защищенный HDCP, не разрешен; • значение 0b01: зарезервировано, **хосты ECI** должны интерпретировать это значение как равное 0b00; • значение 0b10: если версия HDCP равна 2.2 или выше, вывод данных разрешен; если версия HDCP ниже 2.2, вывод данных не разрешен; • значение 0b11: любой вывод данных, защищенный HDCP, разрешен. "HDCP 2.2 или выше" означает запрет использовать для контента приложения HDCP версии ниже 2.2, т. е. запрет вывода на ретрансляторы стандартов HDCP1.x, HDCP2.0 и HDCP2.1, а также на устройства стандартов HDCP1.x. См. определение "потока контента типа 1" в [b-HDCP2.3]
OcWm	5	Если значение этого бита равно 0b1, вывод декодированного элемента контента разрешен только с применением водяного знака, внесенного оборудованием **CPE** в соответствующий элемент контента. См. примечание 3
OcDtcp	6,7	Все выходные данные, защищенные DTCP При OcAnyOther, равном 0b0: • значение 0b00: выходные данные, защищенные DTCP, разрешены; • значение 0b01: если версия DTCP ниже 2, вывод данных не разрешен, если версия DTCP равна 2 или выше, вывод данных разрешен; • значение 0b10: зарезервировано; значение не допускается. **Хосты ECI** должны интерпретировать это значение как равное 0b11; • значение 0b11: вывод данных, защищенный DTCP, не разрешен. При OcAnyOther, равном 0b1: • значение 0b00: вывод данных, защищенный DTCP, не разрешен; • значение 0b01: зарезервировано, **хосты ECI** должны интерпретировать это значение как равное 0b00; • значение 0b10: если версия DTCP равна 2 или выше, вывод данных разрешен; если версия DTCP ниже 2, вывод данных не разрешен; • значение 0b11: любой вывод данных, защищенный DTCP, разрешен
OCDownResHDCP1	8	Вывод данных, защищенный HDCP1.x, разрешен, если значение поля **OcHdcp** равно 0b01 и контент сжат до 720 p или менее, если значение этого поля равно 0b0; вывод данных не разрешен, если значение этого поля равно 0b1
reserved	9–13	Значение данного поля должно быть задано равным 0b00000 **клиентами ECI**, соответствующими данной версии спецификации. Реализация **хоста ECI**, соответствующего данной версии спецификации, может игнорироваться этим полем

Таблица 9.8.2.6.1-1 – Спецификация вектора управления выходными данными

ПРИМЕЧАНИЕ 1. –	Управление аналоговыми выходными данными успешно обеспечивается стандартными полями URI **dot** и **ict**.
ПРИМЕЧАНИЕ 2. –	OcAnyOther успешно переносит поле управления выходными данными из черного списка выходных данных (значение равно 0b0) в белый список выходных данных (значение равно 0b1). Если поле выходных данных равно 0b1, это означает, что оно внесено в список.
ПРИМЕЧАНИЕ 3. –	Может потребоваться утверждение систем защиты водяными знаками, подходящих для этого применения. Хосты ECI, поддерживающие широковещательную и многоадресную передачу, должны поддерживать защиту водяными знаками. В рамках определения применения системы защиты водяными знаками к оборудованию CPE на базе ECI должна быть предусмотрена возможность уникальной идентификации набора микросхем (например, путем извлечения идентификатора набора микросхем из водяного знака).

Если к выходным данным применяются несколько полей ocVector (например, вывод данных через IP, защищенный системой DTCP-IP), должны действовать наиболее жесткие ограничения.

Подробная семантика

- **Клиент ECI** работает в режиме дешифрования.

9.8.2.6.2 Сообщение getEncrOutputCtrl

C→H **uint getEncrOutputCtrl(**ushort **mh,** byte **keyId**[MaxUuidLen])

- Это сообщение получает управление выходными данными для будущего контента.

Определение свойства

- Управление выходными данными определено в таблице 9.8.2.6.1-1.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.

9.8.2.7 API маркирования водяными знаками

9.8.2.7.1 Общие сведения

API маркирования позволяет **клиентам ECI** обнаруживать встроенные системы маркирования водяными знаками, доступные через **хост ECI,** а затем вступать в диалог настройки управления с такими системами. Системы маркирования могут быть способны вступать в диалог только с ограниченным числом **клиентов ECI**, а также маркировать только ограниченное количество сеансов **указателей медиаданных** одновременно.

Системы маркирования могут выражать намерение взаимодействовать с авторизованными **клиентами ECI**. Такого рода авторизация может, помимо прочего, выполняться с использованием сообщений setMarkMeta и getMarkMeta и диалога авторизации, определяемого системой маркирования.

Клиенты ECI могут резервировать доступ к системе маркирования при успешном завершении диалога взаимодействия. **Клиент ECI** (с соответствующим идентификатором) продолжает взаимодействовать с системой маркирования до тех пор, пока он не будет удален из устройства **CPE,** либо до прекращения взаимодействия.

9.8.2.7.2 Сообщение getDcrMarkSyst

C→H MarkSystDescr **getDcrMarkSyst()**

- Это сообщение позволяет **клиенту ECI** считывать дескрипторы для доступных систем маркирования.

Определение свойства

Тип результата MarkSystDescr должен соответствовать определению, приведенному в таблице 9.8.2.7.2-1.

Таблица 9.8.2.7.2-1 – Определение типа MarkSystDescr

```
#define MaxMarkSystDescr 16;

typedef ushort MarkId;  /* Идентификатор маркирования ECI, присвоенный системе маркирования */
//Значения markId: значения 0x8xxx используются для проприетарных систем маркирования.
//                0x0000 означает, что система маркирования отсутствует
//                Остальные значения зарезервированы интерфейсом ECI,
//                назначение новых идентификаторов и их публикация определены в других
источниках.

typedef struct MarkSystDescrElem {
    MarkID markId;        /* Идентификатор системы маркирования */
    uchar  nrClients;     /* количество клиентов, которые могут поддерживаться в данный момент*/
    uchar   markSystFlags  /* поле определено ниже */
} MarkSystDescr[MaxMarkSystDescr];
// Все доступные системы маркирования перечисляются в качестве первых элементов
// дескриптора MarkSystDescr. Остальные элементы используют markId==0x0000.


// markSystFlags:
//   бит 0 сообщает, что авторизация необходима (0b1) или не требуется (0b0)
//   бит 1  сообщает о наличии (0b1) или отсутствии (0b0) поддержки скремблированного потока
//   бит 2 сообщает о наличии (0b1) или отсутствии (0b0) поддержки
//   нескольких одновременных потоков
//   другие биты зарезервированы и должны игнорироваться клиентами
//   в соответствии с положениями настоящей Рекомендации
```

9.8.2.7.3 Сообщение setDcrMarkMeta

C→H setDcrMarkMeta(MarkID **markId,** uchar **index,** byte **data[32])**

- **Это сообщение позволяет хосту ECI** устанавливать метаданные управления для системы маркирования.

Определение параметра

markId: MarkID	Идентификатор системы маркирования, для которого устанавливается определение свойства
index: uchar	Субсвойство, устанавливаемое для систем маркирования
data[32]: byte	Значение, применяемое к субсвойству, обозначенному индексом

9.8.2.7.4 Сообщение getDcrMarkMeta

C→H byte[32] getDcrMarkMeta(MarkID **markId**, uchar **index)**

- Это сообщение позволяет **клиенту ECI** получать метаданные управления для системы маркирования.

Определение свойства

- Метаданные для системы индексов субсвойств с идентификатором **markID**.

Определение параметра

markId: MarkID	Идентификатор системы маркирования, для которого считывается определение свойств: тип результата MarkSystDescr должен соответствовать определению, приведенному в таблице 9.8.2.7.4-1
index: uchar	Субсвойство системы маркирования для считывания

9.8.2.7.5 Сообщение setDcrMarkBasic

C→H setDcrMarkBasic(ushort **mH,** byte **keyId**[MaxUuidLen], MarkID **markId**, byte **data**[16])

- Это сообщение позволяет **клиенту ECI** устанавливать максимальное количество данных (128 бит), используемых для маркирования контента, подлежащего дескремблированию с назначенным ключом.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
markId: MarkID	Идентификатор системы маркирования
data[16]**:** byte	128-битное значение

9.8.2.7.6 Сообщение setDcrMarkExt

C→H setDcrMarkExt(ushort **mH,** byte **keyId**[MaxUuidLen], ushort **markId**, uint **dataLen**, byte **data**[]**)**

- **Это сообщение позволяет клиенту ECI** устанавливать расширенные полезные данные для системы маркирования контента, подлежащего дескремблированию с использованием назначенного ключа.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId: byte[MaxUuidLen]	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
markId: ushort	Идентификатор системы маркирования, используемой для маркирования контента
datalen: uint	Длина поля данных
Data[]**:** byte	Полезные данные для системы маркирования

9.8.2.8 API родительского контроля

9.8.2.8.1 Сообщение setDcrParCtl

C→H setDcrParCtl(ushort **mH,** byte **keyId**[MaxUuidLen], ParCond **pC)**

- **Это сообщение позволяет клиенту ECI** устанавливать условия родительского рейтинга (**pC**) для контента mH, подлежащего дескремблированию с использованием назначенного ключа.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому условие родительского контроля **pC** применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления
pC: ParCond	Условия родительского контроля, применяемые к контенту. В таблице 9.8.2.8.1-1 приведено определение ParCond

Таблица 9.8.2.8.1-1 – Спецификация типов условий родительского контроля

```
typdef struct ParCond {
    byte basicCondition; /* см. таблицу 9.8.2.8.1-2 */
    byte extendedQualifier[16];
} ParCond;
```

Таблица 9.8.2.8.1-2 – Определение базового условия родительского контроля

Название	Биты	Описание
AuthRequired	7	0b1: перед рендерингом контента требуется родительская аутентификация. 0b0: в зависимости от значения extendedQualifier может потребоваться родительская аутентификация
ToggleBit	6	Бит чередуется в потоке, указывая тем самым на новое требование родительской аутентификации при изменении значения бита
Reserved	4,5	Задается равным 0b00
QualifierFormat	0..3	Отображает формат поля extendedQualifier. Значение 0x0 указывает, что "значение не задано"; значение поля ExtendedQualifier устанавливается равным 0; значение 0x1 указывает, что поле ExtendedQualifier содержит дескриптор родительского контроля DVB, как указано в [ETSI EN 300 468]. Остальные байты должны равняться нулю. Родительская аутентификация требуется даже при AuthRequired == 0b0, если требуемый для заданной страны рейтинг превышает предел, установленный родительским объектом (как определено семантикой дескриптора родительского рейтинга DVB). Значения 0x2..0xF зарезервированы для использования в будущем

Подробная семантика

- **ECI** позволяет передавать условия аутентификации родительского рейтинга вместе с контентом в качестве обязательства системе, защищающей дескремблированный контент.
- **Клиент ECI** работает в режиме дешифрования.

9.8.2.8.2 Сообщение getEncrParCtrl

C→H ParCond getEncrParCtrl(ushort **mh,** byte **keyId**[MaxUuidLen])

- Это сообщение позволяет **клиенту ECI** получать условие родительского контроля для будущего контента.

Определение свойства

- URI родительского контроля определена в таблице 9.8.2.8.1-2.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.

9.8.2.9 API синхронизации свойств управления

9.8.2.9.1 Сообщение setCpSync

C→H setCpSync(ushort **mH,** byte **keyId**[MaxUuidLen]**)**

- Это сообщение уведомляет **хост ECI**, что секция будущего контента, обозначенная идентификатором keyId, обладает свойствами контента, установленными посредством

стандартной URI, пользовательской URI, базовой URI, интерфейсами API управления выходными данными, системы защиты водяными знаками и родительского контроля.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому условие родительского контроля **pC** применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления

Подробная семантика

- Сообщение инициирует осуществляемую **хостом ECI** надлежащую подготовку к предстоящим изменениям в свойствах контента. Подготовка включает отправку сообщения reqCpChange любому **микросерверу** с установленным **соединением импорта/экспорта** с этим сеансом **указателя медиаданных**.
- **Клиент ECI** работает в режиме дешифрования.

9.8.2.9.2 Сообщение reqCpChange

H→C reqCpChange(ushort **mh,** byte **keyId**[MaxUuidLen])

- Это сообщение инициирует осуществляемую **микросервером** подготовку к изменению свойства контента на основе последних планируемых значений свойств дешифрованного контента, который повторно шифруется **микросервером**.

Определение свойства

- URI родительского контроля определена в таблице 9.8.2.8.1-2.

Определение параметра

mH: ushort	**Указатель медиаданных** контента, подлежащего шифрованию
keyId[MaxUuidLen]**:** byte	KeyID представляет собой идентификатор UUID в сетевом порядке байтов, к которому URI применяется при декодировании в файловом формате; при этом байт 0 содержит значение 0x00 (четное) или 0x01 (нечетное) для потоков в формате TS, указывая на возможность применения к следующему слову управления

Подробная семантика

- **Клиент ECI** работает в режиме шифрования.
- **Клиент ECI** принимает свойства будущего контента, относящегося к KeyId, в дешифрованном потоке и подготавливает обновленные настройки шифрования для нового контента (для которого может потребоваться новое слово управления).

9.8.2.10 API родительской аутентификации

9.8.2.10.1 Общие сведения

Аутентификация для родительского утверждения может быть выполнена непосредственно **клиентом ECI** во время сеанса MMI. Как вариант, **клиент ECI** может посылать **хосту ECI** запрос на выполнение родительской аутентификации для согласования управления PIN-кодом, а также для усовершенствования **пользовательского** интерфейса посредством естественной интеграции запросов PIN-кода в **пользовательский** интерфейс **хоста ECI**. В свою очередь, **пользователь** через **хост ECI** может выбирать **клиента ECI** из имеющихся вариантов для проведения родительской аутентификаци с использованием сообщения ParAuthDel API для делегирования родительской аутентификации, как указано в пункте 9.8.2.11. Это может быть целесообразным при условии, что **клиент ECI**, обрабатывающий значительное число элементов контента, не может делегировать собственную родительскую аутентификацию, но может выполнять родительскую аутентификацию от имени **хоста ECI**.

Кроме того, данный API позволяет **клиенту ECI** инициировать родительскую аутентификацию элемента контента до открытия медиасеанса, например в целях родительской аутентификации предстоящего события записи.

9.8.2.10.2 Стандартная функция родительской аутентификации

В настоящем разделе определен набор требований к стандартной функции родительского рейтинга, основанной на 4-символьных PIN-кодах. **Хост ECI** должен быть способен выполнять данную функцию по запросу **клиента ECI**, либо данная функция выполняется **клиентом ECI** от имени **хоста ECI**, если **хост ECI** предлагает такую услугу через API делегирования родительской аутентификации.

Хост ECI или **клиент ECI** могут выполнять альтернативную функцию аутентификации, отличную от описанной далее в настоящем разделе, при условии, что подобная функция обеспечивает по меньшей мере целостность родительской аутентификации механизма, определенного в настоящем разделе.

Следующие функциональные возможности применимы к стандартному механизму родительской аутентификации, основанной на PIN-коде:

1) Родительская аутентификация основана на применении PIN-кода, состоящего не менее чем из 4 буквенно-цифровых символов из минимального набора, представленного не менее чем 10 символами (например, цифрами).

2) Настройка PIN-кода должна быть защищена самим PIN-кодом или главным механизмом аутентификации, который блокирует доступ к ресурсам или службам, связанным с материальными ценностями, доступ несовершеннолетних к которым крайне нежелателен и должен быть строго ограничен.

3) Все применимые пределы родительского рейтинга должны быть защищены при помощи PIN-кода или главного механизма аутентификации согласно пункту 2) выше.

4) Требования к планируемому главному механизму аутентификации должны, как минимум, обеспечивать целостность аутентификации с помощью PIN-кода, определенной в настоящем разделе и применяемой без помощи главного механизма аутентификации.

5) При приобретении хоста исходный PIN-код родительского рейтинга или средства аутентификации посредством главного механизма аутентификации передаются только владельцу.

6) При установке нового клиента **оператор** передает исходный PIN-код или средства аутентификации посредством главного механизма аутентификации только владельцу.

7) **Производитель** или лицо, ответственное за хранение и действующее от его имени, могут предоставить механизм для сброса PIN-кода до исходного значения или услугу, с помощью которой владелец может установить новое значение PIN-кода, которое будет передано только владельцу.

8) **Оператор** может предоставить механизм для сброса PIN-кода до исходного значения или услугу, с помощью которой владелец может установить новое значение PIN-кода, которое будет передано только владельцу.

9) При пяти последовательных неудачных попытках аутентификации в течение 15 минут функция родительской аутентификации отказывает в выполнении новой аутентификации в течение как минимум 15 минут.

10) Должна быть отключена возможность восстановления или сброса PIN-кода с помощью стандартных **пользовательских** приложений, а также загруженных приложений, запущенных на оборудовании **CPE** или каких-либо **пользовательских** или стандартных интерфейсах.

9.8.2.10.3 Сообщение reqParAuthChk

C→H reqParAuthChk(ushort **mH)** →
C→H resParAuthChk(ushort **mH,** bool **ok)**

- Это сообщение позволяет **клиенту ECI** посылать **хосту ECI** запрос на выполнение проверки родительской аутентификации с использованием стандартной функции родительской аутентификации (см. пункт 9.8.2.10) и на возврат результата в сообщении-отклике.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
ok: bool	Значение *true* соответствует успешно выполненной аутентификации, (значение *false*) указывает на невыполнение аутентификации, включая режим ожидания

Подробная семантика

- **Хост ECI** распознает только одну невыполненную проверку родительской аутентификации для каждого **указателя медиаданных**. Отправка второго запроса на тот же **указатель медиаданных** до получения отклика на предыдущий запрос или его отмены приведет к двум одинаковым **откликам**.
- **reqParAuthChk. Хост ECI** должен использовать значение времени ожидания при отправке запроса родительской аутентификации, которое завершится в течение обоснованного периода, если лицо отсутствует или не желает провести аутентификацию, как предлагается в [b-ITU-T J Suppl. 7].

9.8.2.10.4 Сообщение reqParAuthChkCan

C→H reqParAuthChkCan(ushort **mH) →**
H→C resParAuthChkCan(ushort **mH)**

- **Клиент ECI** отменяет все предыдущие запросы интерфейсу **ECI** на выполнение родительской аутентификации.

Определение параметра запроса

mH: ushort	**Указатель медиаданных контента**, подлежащего декодированию

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию

Постусловия (отклик)

1) Отклик на предыдущее сообщение **reqParAuthChk может** быть возвращен **хостом ECI клиенту ECI** перед получением сообщения **resParAuthChkCan**, но не после него.

9.8.2.10.5 Сообщение reqParAuthCid

H→C reqParAuthCid(uint **cidLength,** byte **cid**[]**) →**
C→H resParAuthCid(bool **ok)**

- Это сообщение **позволяет хосту ECI** отправлять **клиенту ECI** запрос на выполнение любой необходимой аутентификации для будущего элемента контента, обозначенного идентификатором **cid**.

Определение параметра запроса

cidLength: uint	Длина параметра cid
cid[]**:** byte	Идентификация контента, подлежащего родительской аутентификации (при необходимости). Первый байт обозначает формат параметра идентификации контента, как указано в таблице9.8.2.10.5-1

Таблица 9.8.2.10.5-1 – Форматы идентификации контента

Название	Значение	Описание
CidDvbEvent	0x01	Идентификация событий DVB. Байты, следующие за байтами в cid, имеют значение последовательности: идентификатор исходной сети (2 байта), идентификатор транспортного потока (2 байта), служебный идентификатор (2 байта), идентификатор события (2 байта), как указано в таблице EIT, приведенной в [ETSI EN 300 468]. Все 2-байтовые поля последовательности представлены в сетевом порядке (первым следует самый старший двоичный разряд)
RFU	прочее	Зарезервировано для использования в будущем

Определения параметра отклика

ok: bool	Значение true указывает на то, что родительская аутентификация проведена успешно или не потребовалась

Подробная семантика

- **Клиент ECI** поддерживает запись в энергонезависимую память идентификационных данных контента, которые аутентифицированы с помощью этой функции. Он может удалить самые ранние и не нужные более записи при недостатке пространства для хранения данных. Минимальные требования к буферизации идентификационных данных контента предлагаются [b-ITU-T J Suppl. 7].

Соответствующие коды ошибок перечислены в таблице 9.8.2.10.5-2.

Таблица 9.8.2.10.5-2 – Коды ошибок API медиасеанса для медиаданных транспортных потоков

Название	Значение	Описание
ErrParAuthCidUnknOk	1	Определение статуса родительской аутентификации элемента контента выполнить не удалось, однако родительская аутентификация проведена и признана корректной

Вышеуказанные состояния ошибки также могут возвращаться в случае отсутствия доступа к необходимым сетевым ресурсам.

9.8.2.11 API делегирования родительской аутентификации

9.8.2.11.1 Общие сведения

API позволяет **клиенту ECI** информировать о том, что он может выполнять стандартную функцию родительской аутентификации, как указано в пункте 9.8.2.10.2, и делегировать **хосту ECI** полномочия для проверки PIN-кода у такого **клиента ECI**.

Клиент ECI может сообщать о поддержке API делегированной аутентификации с использованием API конфигурации во время инициализации **клиента ECI**.

ПРИМЕЧАНИЕ. – В то же время клиент ECI может отказаться от делегирования собственной родительской аутентификации, например, по коммерческим, юридическим причинам или по соображениям безопасности.

Хост ECI предоставляет функцию настройки, позволяющую пользователю выбирать **хост ECI** для стандартной аутентификации родительского контроля или делегировать стандартную аутентификацию родительского контроля одному из **клиентов ECI**, предоставляющему эту функцию.

9.8.2.11.2 Сообщение reqParAuthDel

H→C reqParAuthDel(ushort **mh**) →
C→H resParAuthDel(ushort **mH,** bool **ok)**

- Это сообщение позволяет **хосту ECI** отправлять **клиенту ECI** запрос на выполнение делегированной родительской аутентификации от его имени для контента при mH.

Определение параметра запроса

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию
ok: bool	Значение *true* указывает на успешную родительскую аутентификацию, значение *false* – на сбой аутентификации или переход в режим ожидания

Подробная семантика

- **Клиент ECI** распознает только одну невыполненную проверку родительской аутентификации для каждого **указателя медиаданных.** Отправка второго запроса на тот же **указатель медиаданных** до получения отклика или отмены предыдущего запроса приведет к двум одинаковым откликам.
- **Клиент ECI** должен использовать значение времени ожидания при отправке запроса на родительскую аутентификацию, которое завершится в течение обоснованного периода, если лицо отсутствует или не желает провести аутентификацию, как предложено в [b-ITU-T J Suppl. 7].

9.8.2.11.3 Сообщение setParAuthDelCan

H→C reqParAuthDelCan(ushort **mH**) →
C→H resParAuthDelCan(ushort **mH)**

- Это сообщение позволяет **хосту ECI** отменять запрос о делегировании родительской аутентификации.

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию

Определение параметра отклика

mH: ushort	**Указатель медиаданных** контента, подлежащего декодированию

Постусловия (отклик)

- Отклик на предыдущее сообщение reqParAuthDel может быть возвращен **хостом ECI клиенту ECI** перед получением сообщения resParAuthDelCan, но не после него.

9.8.2.12 API управления системой защиты

9.8.2.12.1 Введение

Контент, дешифрованный **клиентом ECI**, может передаваться на различные выходы оборудования **CPE**. Как правило, выход защищен системой защиты выходных данных. Система защиты выходных данных может предусматривать возможность приема от **клиента ECI** системных сообщений о возобновлении (SRM), а также предлагать **клиенту ECI** возможность блокировки вывода на устройства, соединение с которыми установлено через эту систему, если идентификаторы соответствующих устройств (в контексте системы защиты выходных данных) числятся скомпрометированными.

Система защиты может поддерживать множество выходов.

9.8.2.12.2 Сообщение getProtSystCtrl

C->H getProtSystCtrl()

- Это сообщение позволяет **клиенту ECI** считывать список систем защиты выходных данных, поддерживаемых оборудованием CPE, с указанием их версий и сведений о поддержке SRM (системных сообщений о возобновлении) и услуг блокировки устройств по идентификатору.

Таблица 9.8.2.12.2-1 – Определение массива параметров управления защитой

```
typedef struct ProtCtrlElem {
    ushort  protSysType;    // тип системы защиты согласно таблице sect-2
    uint srmSupp:4;         // уровень поддержки SRM согласно таблице sect-3
    uint devIdSupp:1;       // 0b0 означает отсутствие поддержки услуг блокировки устройств
                            // по идентификатору,
                            // 0b1 означает поддержку услуг блокировки устройств по идентификатору
    uint reserved:11;       // зарезервировано; должно иметь значение 0b00000000000
} ProtCtrlElem;

#define MaxProtCtrlArr 32
typedef ProtCtrlElem ProtCtrlArr[MaxProtCtrlArr];
    // Указанная в массиве система защиты может обеспечивать защиту нескольких выходов.
    // В ProtCtrlArr не допускаются повторяющиеся значения ProtCtrlElem, кроме случая,
    // когда protSustType=0x0000. Все ProtCtrlElem, у которых ProtColElem не равно 0x0000,
    // должны располагаться в ячейках ProtCtrlArr с наименьшими индексами,
    // а те, у которых ProtColElem равно 0x0000 – в конце массива
```

Таблица 9.8.2.12.2-2 – Значения признака типа системы защиты выходных данных

Название	Значение	Тип системы защиты выходных данных
OpNoProtSyst	0x0000	Система защиты выходных данных отсутствует
OpHDCP_1	0x0010	HDCP версии 1
OpHDCP_21	0x0011	HDCP версий 2.0 или 2.1
OpHDCP_22	0x0012	HDCP версии 2.2 или выше
OpDTCP_1	0x0020	DTCP версии 1
OpDTCP_2	0x0021	DTCP версии 2 или выше
OpDTCP_IP1	0x0030	DTCP IP
Proprietary	0x8xxx	Может быть определено вне рамок настоящей спецификации
Reserved	Другие значения	Зарезервировано для использования в будущем

Таблица 9.8.2.12.2-3 – Значения признака поддержки SRM

Название	Значение	Тип системы защиты выходных данных
SrmNone	0x0	Поддержка SRM отсутствует
SrmProtSysSpecV1	0x1	Поддержка SRM в соответствии с версией 1 (но не выше) спецификации системы защиты выходных данных
SrmProtSysSpecV2	0x2	Поддержка SRM в соответствии с версией 2 (но не выше) спецификации системы защиты выходных данных
SrmProtSysSpecV3	0x3	Поддержка SRM в соответствии с версией 3 (но не выше) спецификации системы защиты выходных данных
SrmProtSysSpecV4	0x4	Поддержка SRM в соответствии с версией 4 (но не выше) спецификации системы защиты выходных данных
reserved	0x5..0xC	Зарезервировано для использования в будущем
Proprietary	0xD-0xF	Может быть определено вне рамок настоящей спецификации

Семантика

- Поддержка услуг блокировки устройств по идентификатору означает, что система защиты поддерживает идентификацию и блокировку любых защищенных соединений с устройством с помощью сообщений reqBlockDevId и resBlockDevId.
- Конфигурация функций защиты выходных данных должна быть статической на протяжении "времени жизни" клиента.

9.8.2.12.3 Сообщение reqSrmMsg

C→H **reqSrmMsg**(ushort **protSysType**, uint **srmLen**, byte **srmData**[]) →
H→C **resSrmMsg**()

- Это сообщение позволяет **клиенту ECI** передать SRM в адрес системы защиты заданного типа.

Определение параметра запроса

protSysType[]**:** ushort	Тип системы защиты, которой адресовано данное SRM. Примечание: SRM могут быть адресованы нескольким типам систем защиты, принадлежащих к одному семейству. В таком случае достаточно передать хосту SRM лишь единожды, не повторяя передачу для каждого типа
srmLength: uint	Длина SRM
srmData:byte[]	SRM

Предварительные условия (запрос)

- Сообщение **reqSrmMsg** ранее не передавалось или в ответ на последнее сообщение **reqSrmMsg** принято сообщение **resSrmMsg**.

Подробная семантика

- Хост ECI передает сообщение **resSrmMsg** как можно скорее.

Таблица 9.8.2.12.3-1 – Коды ошибок reqSrmMsg

Название	Описание
ErrReqSrmMsgOverflow	См. пункт 9.8.2.12.7

9.8.2.12.4 Сообщение reqInfoDevId

H→C **reqInfoDevId**(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[])→
C→H **resInfoDevId**(ushort **mh**)

- Это сообщение позволяет **хосту ECI** указать, на какие устройства (идентифицируемые по **devId**) передается контент, который может быть дешифрован устройством, с использованием системы защиты **protSysType** в сеансе дешифрования **mh**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса дешифрования, в котором участвует устройство с идентификатором **devId**
protSysType: ushort	Система защиты выходных данных, используемая для защиты контента, который доставляется на устройство с идентификатором **devId** – см. таблицу 6.4.2-1 в [b-ITU-T J Suppl. 7]
lenDevId: uint	Длина поля **devId** в байтах
devId[]**:** byte	Идентификатор устройства; конкретная кодировка определяется в дополнительной спецификации

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса дешифрования, для которого предназначен отклик

Предварительные условия (запрос)

- Сообщение **reqInfoDevId** ранее не передавалось в сеансе **mh** или в ответ на последнее сообщение **reqInfoDevId** в сеансе **mh** было принято сообщение **resInfoDevId**.

Подробная семантика

- **Хост ECI** передает идентификатор **devId** каждого устройства, подключенного к выходу сеанса **mh**, как можно скорее.

Таблица 9.8.2.12.4-1 – Коды ошибок reqInfoDevId

Название	Описание
ErrReqInfoDevOverflow	См. пункт 9.8.2.12.7

9.8.2.12.5 Сообщение reqBlockDevId

C→H **reqBlockDevId**(ushort **mh**, ushort **protSysType**, uint **lenDevId**, byte **devId**[])→
H→C **resBlockDevId**(ushort **mh**)

- Это сообщение позволяет **клиенту ECI** блокировать по **devId** устройства, на которые передается дешифрованный контент с использованием системы защиты **protSysType** в сеансе дешифрования **mh**.

Определение параметра запроса

mh: ushort	**Указатель медиаданных** сеанса дешифрования, в котором участвует устройство с идентификатором **devId**
protSysType: ushort	Система защиты выходных данных, используемая для защиты контента, который доставляется на устройство с идентификатором **devId** – см. таблицу 6.4.2-1 в [b-ITU-T J Suppl. 7]
lenDevId: uint	Длина поля **devId** в байтах
devId[]**:** byte	Идентификатор устройства; конкретная кодировка определяется в дополнительной спецификации

Определение параметра отклика

mh: ushort	**Указатель медиаданных** сеанса дешифрования, для которого предназначен отклик

Предварительные условия (запрос)

- **Сообщение reqBlockDevId** ранее не передавалось в сеансе **mh** или в ответ на последнее сообщение **reqBlockDevId** в сеансе **mh** было принято сообщение **resBlockDevId**.

Семантика

- Получив действительный **reqBlockDevId**, **хост ECI** возвращает отклик **ErrReqOkNoId** (см. таблицу 9.3.4-1) и обеспечивает блокировку вывода на устройство с идентификатором **devId**.

9.8.2.12.6 Сообщение setBlockProtSyst

C→H setBlockProtSyst(ushort **mh**, ushort **protSysType** bool **block)**

- Это сообщение позволяет **клиенту ECI** блокировать весь дешифрованный контент, передаваемый с использованием системы защиты **protSysType** в сеансе дешифрования **mh**.

Определение параметра

mh: ushort	**Указатель медиаданных** сеанса дешифрования, для которого требуется заблокировать контент
protSysType: ushort	Система защиты выходных данных, используемая для защиты контента, который доставляется на устройство с идентификатором **devId** – см. таблицу 6.4.2-1 в [b-ITU-T J Suppl. 7]
block: bool	Имеет значение **True**, если контент подлежит блокировке, и **False –** в противном случае

Семантика

- Если значение параметра **block** для систем защиты типа **protSysType** в сеансе **mh** меняется с **True** на **False**, идентификаторы **devID** всех устройств, подключенных через систему с этим **protSysType** для вывода в сеансе **mh**, передаются **хостом ECI** в сообщении **reqInfoDevId**, если реализация **protSysType** это допускает (в соответствии с результатом **getProtSystCtrl**).

9.8.2.12.7 Коды ошибок API управления системой защиты

- Коды ошибок API управления системой защиты перечислены в таблице 9.8.2.12.7-1.

Таблица 9.8.2.12.7-1 – Коды ошибок, относящиеся к API управления системой защиты

Название	Значение	Описание
ErrReqSrmMsgOverflow	–256	**Хост ECI** уведомляет, что в данный момент не может принять следующее сообщение ReqSrmMsg
ErrReqInfoDevOverflow	–257	**Клиент ECI** уведомляет, что в данный момент не может принять следующее сообщение ReqInfoDev

9.9 Интерфейсы API для связи между клиентом ECI и приложением

9.9.1 Список интерфейсов API, определенных в настоящем пункте

В таблице 9.9.1-1 приведен список интерфейсов API, рассматриваемых в данном пункте.

Таблица 9.9.1-1 – Интерфейсы API для ресурсов, имеющих отношение к связи между клиентом ECI и приложением

Пункт	Наименование API	Описание
9.9.2	API связи между клиентами	Позволяет **клиенту ECI** устанавливать прямую связь с другим **клиентом ECI**

9.9.2 API связи между клиентами

9.9.2.1 Общие сведения

Хост ECI предоставляет среду стандартного обмена информацией между **клиентами ECI** в форме информации импорта/экспорта, информации URI и контента. **Клиенты ECI** могут связываться друг с другом, что позволяет им обеспечивать дополнительные функциональные возможности (не определенные интерфейсом **ECI** на текущий момент). **Клиенты ECI** могут регистрировать свою принципиальную способность и готовность поддерживать связь между клиентами через ресурс обнаружения (см. пункт 9.4.2). После инициализации системы они могут считывать идентификационные данные других **клиентов ECI**, включая установленное **соединение импорта/экспорта**. **Клиенты ECI** могут открывать канал связи с потенциальным партнером и использовать его для обмена сообщениями. Канал может быть отменен обеими сторонами. Канал **клиента ECI** закрывается **хостом ECI** при остановке или повторной инициализации его партнера – **клиента ECI**.

Хост ECI предоставляет **клиенту ECI** идентификационные данные, которые аутентифицируются с использованием **цепочек сертификатов ECI**, предоставленных с **клиентами ECI**. **Клиенты ECI** предоставляют дополнительный механизм независимой аутентификации в том случае, если связь с партнером может привести к угрозе безопасности.

В том случае, если связь установлена между **клиентом ECI**, выполняющим декодирование контента, и другим **клиентом ECI**, вслед за этим выполняющим повторное шифрование данного контента (**микросервером**), при настройке канала связи рекомендуется, чтобы канал инициировался (открывался) **микросервером**.

В таблице 9.9.2.1-1 приведены сообщения API связи между клиентами.

Таблица 9.9.2.1-1 – Сообщения API связи между клиентами

Сообщение	Тип	Направление	Маркер	Описание
getIccMaxClients	S	C→H	0x0	**Клиент ECI** считывает максимальное число **клиентов ECI**, которое может поддерживать **хост ECI**
reqIccSystemReady	A	H→C	0x1	**Хост ECI** информирует **клиента ECI**, что все **клиенты ECI** инициализированы
getIccClientInfo	S	C→H	0x2	**Клиент ECI** считывает идентификационные данные и статус соединения другого **клиента ECI** в системе
reqIccPipeOpen	A	C→H	0x3	Запрос на открытие канала связи с другим **клиентом ECI**
reqIccPipeOpenReq	A	H→C	0x4	Входящий запрос от другого **клиента ECI** на открытие канала
reqIccPipeCancel	A	C→H	0x5	**Клиент ECI** отменяет канал связи
reqIccPipeClose	A	H→C	0x6	**Хост ECI** информирует **клиента ECI**, что канал связи закрыт
reqIccPipeMsgSend	A	C→H	0x7	**Клиент ECI** отправляет сообщение своему партнеру по каналу связи
reqIccPipeMsgRecv	A	H→C	0x8	**Клиент ECI** получает сообщение от своего партнера по каналу связи

9.9.2.2 Сообщение getIccMaxClients

C→H uint getIccMaxClients()

- Получает максимальное число **клиентов ECI**, которое может поддерживать **хост ECI**.

Определение свойства

- Целое число без знака, представляющее максимальное число **клиентов ECI**, которое может поддерживать **хост ECI**.

9.9.2.3 Сообщение reqIccSystemReady

H→C reqIccSystemReady()

- **Хост ECI** информирует **клиента ECI**, что все другие **клиенты ECI** инициализированы.

Семантика

- Это сообщение передается при инициализации системы и сообщает всем **клиентам ECI**, зарегистрированным на данном интерфейсе API, что можно начинать считывание реестра клиентской информации и пробовать открывать канал связи с другими **клиентами ECI**.
- Поле ConnId в полученном результате отражает последний статус **соединений импорта/экспорта клиента ECI** с потенциальным партнером. Они могут подвергаться изменениям.
- Сообщение с результатом не требуется.

9.9.2.4 Сообщение getIccClientInfo

C→H ClientInfo getIccClientInfo(ushort **clientId)**

- **Клиент ECI** считывает идентификационные данные и статус соединения другого **клиента ECI** в системе.

Определение параметра

clientId: ushort	Идентификатор клиента для настройки каналов. Этот идентификатор не изменяется на протяжении жизненного цикла системы. Он изменяется при повторной инициализации

Определение свойства

- Идентификатор connectionID является динамическим свойством.
- ClientInfo – структура, предоставляющая идентификационные данные назначенного **клиента ECI** и любого **соединения импорта/экспорта** с этим **клиентом ECI**, как определено ниже.

Описание типа для 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];
}
```

Определения полей

operatorId: ECI_Operator_Id	Идентификатор оператора **клиента ECI**
platformOperationId: ECI_Platform_Operation_Id	Идентификатор системы управления платформой **клиента ECI**
client: union	Либо ECI_Client_Series_Id либо ECI_Client_Id. Поле типа clientSeriesId и clientId определяет идентификатор – clientSeriesId или clientId
VendorId: ECI_Vendor_Id	Идентификатор поставщика **клиента ECI**
clientSeriesId: ECI_Client_Sesies_Id	Идентификатор клиентской серии **клиента ECI**
clientId: ECI_Client_Id	Идентификатор клиента **клиента ECI**
connId: ushort[MaxConnId]	Массив идентификаторов соединений; значение 0xFFFF посылает информацию о пустой записи в массиве. Пустые записи размещены в конце архива

9.9.2.5 Сообщение reqIccPipeOpen

C→H reqIccPipeOpen(ushort **clientId,** byte **protocolId[16]) →**
H→C resIccPipeOpen(ushort **clientId)**

- Это сообщение позволяет **клиенту ECI** посылать **хосту ECI** запрос на открытие канала связи с другим **клиентом ECI**.

Определение параметра запроса

clientId: ushort	Идентификатор клиента, для которого запрашивается канал
protocolId[16]: byte	Идентификатор используемого протокола сообщений. Этот идентификатор – UUID [IETF RFC 4122] с октетами в сетевом порядке в массиве

Определение параметра результата

clientId: ushort	Идентификатор клиента, для которого запрашивается открытие канала

Предварительные условия (отклик)

- Канал открыт или возвращается код ошибки. Соответствующие коды ошибок перечислены в таблице 9.9.2.5-1.

Таблица 9.9.2.5-1 – Коды ошибок reqIccPipeOpen

Название	Описание
ErrIccPipeOpenReject	См. таблицу 9.9.2.11-1
ErrIccPipeOpenNoConn	
ErrIccPipeOpenProtocol	
ErrIccPipeOpenNotReady	

9.9.2.6 Сообщение reqIccPipeOpenReq

H→C reqIccPipeOpenReq(ushort **clientId,** byte **protocolId**[16]**) →**
C→H resIccPipeOpen(ushort **clientId)**

- Это сообщение позволяет **клиенту ECI** получать входящий запрос от другого **клиента ECI** на открытие канала через **хост ECI**.

Определение параметра запроса

clientId: ushort	Идентификатор клиента, запрашивающего канал
protocolId[16]**:** byte	Идентификатор используемого протокола сообщений. Этот идентификатор – UUID [IETF RFC 4122] с октетами байтов в сетевом порядке

Определение параметра результата

clientId: ushort	Идентификатор клиента, запросившего канал

Семантика

- Значение отклика идентификатора clientId должно быть идентичным значению запроса.

Предварительные условия (отклик)

- **Клиент ECI** может отказаться от канала. Коды ошибок равны кодам для открытия канала и транспарентным образом передаются отправителю запроса. Они перечислены в таблице 9.9.2.5-1.

9.9.2.7 Сообщение reqIccPipeCancel

C→H reqIccPipeCancel(ushort **clientId) →**
H→C resIccPipeCancel(ushort **clientId)**

- Это сообщение позволяет **клиенту ECI** уведомлять **хост ECI** о намерении закрыть канал связи.

Определение параметра запроса

clientId: ushort	Идентификатор клиента канала, который был отменен

Определение параметра(ов) результата

clientId: ushort	Идентификатор клиента канала, который был отменен

Семантика

- Значение отклика идентификатора clientId должно быть идентичным значению запроса.

Предварительные условия (отклик)

- Работа канала связи прекращена: **клиент ECI**, запрашивающий отмену канала, не будет больше получать сообщений от канала.

Подробная семантика

- Если канал не был открыт, при обработке сообщения ошибка не возникает.

9.9.2.8 Сообщение qIccPipeClose

H→C reqIccPipeClose(ushort **clientId,** uint **reason) →**
C→H resIccPipeClose(ushort **clientId)**

- Это сообщение позволяет **хосту ECI** информировать **клиента ECI**, что канал связи с партнером закрыт.

Определение параметра запроса

clientId: ushort	Идентификатор клиента канала, который был закрыт
reason: uint	Обоснование закрытия канала. Значения перечислены в таблице 9.9.2.11-1

Таблица 9.9.2.8-1 – Значения для обоснования reqIccPipeClose

Название	Значение	Описание
IccPipeCloseCancel	0x01	Канал закрыт партнером с использованием сообщения reqIccPipeCancel
IccPipeCloseStop	0x02	Канал закрыт **хостом ECI** как следствие прекращения работы партнера – **клиента ECI**. Есть вероятность, что **клиент ECI** впоследствии инициализирован повторно
RFU	Прочее	Зарезервировано для использования в будущем

Определение параметра результата

clientId: ushort	Идентификатор клиента, канал которого был закрыт

Предварительные условия (запрос)

- Сообщения через канал больше отправляться не будут.

Предварительные условия (отклик)

- **Клиент ECI** не будет пытаться посылать новые сообщения по (закрытому) каналу.

9.9.2.9 Сообщение reqIccPipeMsgSend

C→H reqIccPipeMsgSend(ushort **clientId,** uint **msgId,** uint **dataLen,** byte **data[])→**
H→C resIccPipeMsgSend(ushort **clientId)**

- Это сообщение позволяет **клиенту ECI** посылать сообщение партнеру по каналу связи. Соответствующие коды ошибок перечислены в таблице 9.9.2.11-1.

Определение параметра запроса

clientId: ushort	Идентификатор клиента, которому отправлено сообщение
msgId: uint	Идентификатор сообщения. Все отрицательные и нулевое значения зарезервированы; все положительные значения определяются конкретным приложением (значение определяется в контексте отправителя и получателя)
dataLen: uint	Длина параметра данных в количестве байтов. Это значение не превышает 32 768
data[]: byte	Поле данных для сообщения

Определение параметра результата

clientId: ushort	Идентификатор клиента канала

Предварительные условия (запрос)

- Следующее сообщение reqIccMsgSend может быть отправлено только после того, как получено предыдущее сообщение resIccMsgSend для того же канала связи.

Таблица 9.9.2.9-1 – Коды ошибок reqIccPipeMsgSend

Название	Описание
ErrIccPipeClosed	См. таблицу 9.9.2.11-1

9.9.2.10 Сообщение reqIccPipeMsgRecv

H-→C reqIccPipeMsgRecv(ushort **clientId,** uint **msgId,** uint **dataLen,** byte **data[])→**
C→H resIccPipeMsgRecv(ushort **clientId)**

- Сообщение позволяет **клиенту ECI** получать сообщения от партнера по каналу связи.

Определение параметра запроса

clientId: ushort	Идентификатор клиента, от которого было получено сообщение
msgId: uint	Идентификатор сообщения. Все отрицательные и нулевое значения зарезервированы; все положительные значения определяются конкретным приложением (значение определяется в контексте отправителя и получателя)
dataLen: uint	Длина параметра данных в количестве байтов. Это значение не превышает 32 768
data: byte[]	Поле данных для сообщения

Определение параметра результата

clientId: ushort	Идентификатор клиента канала

Предварительные условия (запрос)

- Следующее сообщение reqIccMsgRecv отправляется только после того, как получено предыдущее сообщение reqIccMsgRecv для того же канала связи.

9.9.2.11 Коды ошибок для связи между клиентами

Коды ошибок для API связи между клиентами перечислены в таблице 9.9.2.11-1.

Таблица 9.9.2.11-1 – Коды ошибок для связи между клиентами

Название	Значение	Описание
ErrIccPipeOpenReject	–256	Партнер отклонил канал связи
ErrIccPipeOpenNoConn	–257	Партнер отклоняет канал связи по причине отсутствия установленного **соединения импорта/экспорта** с клиентом ECI
ErrIccPipeOpenProtocol	–258	Партнер отклоняет протокол, предложенный для канала связи
ErrIccPipeOpenNotReady	–259	Партнер не находится в состоянии готовности принять связь по каналу. Целесообразно повторить попытку установления канала связи позже
ErrIccPipeClosed	–260	Канал связи закрыт

10 Обязательные и дополнительные функциональные возможности хоста ECI

10.1 Введение

Технические спецификации системы **ECI** поддерживают технические решения для широкого спектра устройств **CPE**, предназначенных для передачи мультимедийных данных. **Производитель оборудования CPE** самостоятельно принимает решение о внедрении в устройства тех или иных функций внешнего интерфейса, основных и серверных функций. Для реализации функций внешнего интерфейса и серверных функций **производитель**, скорее всего, будет внедрять только те интерфейсы API **ECI**, которые соответствуют его аппаратному стеку и стеку протоколов. В таблице 10.2-1 приведен перечень обязательных (m), дополнительных (o) и условных (c) интерфейсов API для различных типов устройств **CPE**, предоставляющий **пользователям** широкие возможности.

10.2 Перечень обязательных и дополнительных функциональных возможностей ECI для различных типов устройства CPE

В таблице 10.2-1 приведен список обязательных и дополнительных функциональных возможностей **ECI** для различных типов устройств **CPE**. Внедрение некоторых интерфейсов API обусловлено наличием тех или иных программно-аппаратных компонентов в конкретном устройстве **CPE**.

Таблица 10.2-1 – Список обязательных и дополнительных функциональных возможностей ECI

API	Пункт	Хост	Условие (если применимо)	Клиент дешифро-вания	Микро-сервер	Микро-клиент
Обнаружение интерфейса хоста	9.4.2	M		M	M	M
MMI	9.4.3	M		O	O	O
IP	9.4.4	C	Если поддерживается подключение по IP	O	O	O
HTTP(S)	9.4.4.6	M		O	O	O
Файловая система	9.4.5	M		O	O	O
Таймер и часы	9.4.6	M		O	O	O
Управление электропитанием	9.4.7	M		O	O	O
Языковые и страновые настройки	9.4.8	M		O	O	O
Усовершенствованная система безопасности (общий)	9.5.2.2	M		M	M	M
API дешифрования усовершенствованной системы безопасности	9.5.2.3	M		M	н/д	M
Экспорт усовершенствованной системы безопасности	9.5.2.4	C	Для записи или шлюза	O	н/д	O
Шифрование усовершенствованной системы безопасности	9.5.2.5	C	Для записи или шлюза	н/д	M	н/д
Смарт-карта	9.5.3	C	Для поддерживаемых устройств чтения смарт-карт	O	O	O
Карусель передачи данных	9.5.4	C	Для радиовещательной сети	O	O	O
Дешифрование (см. примечание)	9.6.2	M		M	н/д	M
Соединение экспорта	9.7.2.3	C	Для записи или шлюза	O	н/д.	O
Соединение импорта	9.7.2.4	C	Для записи или шлюза	н/д	M	н/д
Повторное шифрование (см. примечание)	9.7.2.5	C	Для записи или шлюза	н/д	M	н/д
Дешифрование микроклиента	9.7.2.6	M		O	н/д	M
Языковые и страновые настройки	9.4.8	M		O	O	O
Стандартная URI	9.8.2.3	M		M	M	M
Пользовательская URI	9.8.2.4	M		M	M	M
Базовая URI	9.8.2.5	M		M	M	M
Управление выходными данными	9.8.2.6	M		M	M	M
Защита водяными знаками	9.8.2.7	C	Для устройств, поддерживающих широковещательную и многоадресную передачу	O	н/д	O
Родительский контроль	9.8.2.8	M		M/O	M/O	M/O
Синхронизация свойств контента	9.8.2.9	M		M	M	M
Родительская аутентификация	9.8.2.10	M		O	н/д	O
Делегирование родительской аутентификации	9.8.2.11	M		O	н/д	O

Таблица 10.2-1 – Список обязательных и дополнительных функциональных возможностей ECI

Связь между клиентами	9.9.2	M		O	O	O

ПРИМЕЧАНИЕ. – Сегменты могут быть предназначены специально для **микросерверов** и клиентов дешифрования. Сам по себе сегмент технически идентичен, однако требуемые ресурсы системы **AS** и соответствующие функции дескремблирования различаются.

В API-интерфейсах обнаружения не предложен механизм, позволяющий **хосту ECI** обнаруживать способность **клиента ECI** дешифровать или шифровать файл и/или медиаданные в формате транспортного потока. Такого рода сигнализация обеспечивается полем mhType параметра decryptId сообщения setDcrMhMatch (см. пункт 9.6.2.2.2). Для повторного шифрования такое обнаружение обеспечивается параметром EciEncrModes сообщения setEncrModes (см. пункт 9.7.2.5.3).

- В устройстве, ориентированном только на потребление и поддерживающем интерфейс **ECI**, должны присутствовать как минимум 2 копии VM и AS-сегменты.
- **Хосты ECI**, поддерживающие функции PVR, должны поддерживать как минимум один дополнительный контейнер (копию **VM**) и **AS-сегмент** для **микросервера**. Если такой **хост ECI** предоставляет также набор функций для воспроизведения сохраненного контента, он должен поддерживать как минимум один дополнительный контейнер (копию VM) и **AS**-сегмент для **микроклиента**, который может декодировать повторно зашифрованный контент.
- **Хосты ECI**, поддерживающие функции сетевого шлюза, должны поддерживать как минимум один дополнительный контейнер (копию **VM**) и AS-сегмент для **микросервера**.

Приложение A

Криптографические функции хоста ECI

(Данное Приложение является неотъемлемой частью настоящей Рекомендации.)

A.1 Хеш-функция

Все хеш-функции, приведенные в настоящей Рекомендации, основаны на SHA256, как указано в [NIST FIPS 197].

Функция *hash* (пункт 5.2) равна SHA-256 (), как указано в [NIST FIPS 197].

В функции кодирования (c-function) asHash (uchar *data, uint datalength, resultLength, uchar *result) используются октеты, начиная с параметра длины dataLength, в качестве *dataIn* octetstring, и вычисляется octetstring resultOut как октетная строка resultLength/8, а затем строка сохраняется в результате согласно формуле:

resultOut = BS2OSP(truncate(SHA-256(OS2BSP(*dataIn*)),resultLength)))

Значение resultLength должно быть кратным 8. Функция truncate выполняет усечение битовой строки слева (параметр 1) до длины в битах, соответствующей параметру 2.

BS2OSP и OS2BSP – функции, преобразующие битовую строку в октетную строку и обратно. Этот процесс описывается в разделе 9 [ITU-T J.1014].

A.2 Асимметричная криптография

Операции асимметричного шифрования и дешифрования описываются в пункте 12.4 [ITU-T J.1014].

A.3 Симметричная криптография

Криптография AES, определенная в настоящей Рекомендации, должна соответствовать указанной в [NIST FIPS 197], если не приведена конкретная ссылка на приложение AES.

Приложения CBC стандарта AES должны соответствовать указанным в [NIST Block 2001], если не приведена конкретная ссылка на приложение для CBC (стандарт AES). Если не указано иное, используется вектор инициализации 0.

Приложения CTR стандарта AES должны соответствовать указанным в [NIST Block 2001], если не приведена конкретная ссылка на приложение для CTR (стандарт AES). Если не указано иное, используется вектор инициализации 0.

A.4 Генерирование случайных чисел

Генерирование случайных чисел, как указано в настоящей Рекомендации, должно соответствовать спецификации, установленной в Приложении A к [ITU-T J.1014].

Приложение B

Параметры функциональной совместимости

(Данное Приложение является неотъемлемой частью настоящей Рекомендации.)

B.1 Введение

В этом приложении описываются параметры, связанные с требованиями к ресурсам для оборудования **CPE**. Соблюдение этих требований обеспечивает функциональную совместимость между **клиентами ECI**, устройствами **CPE** и услугами безопасности **ECI**, предоставляемыми сетями.

B.2 Длина списка аннулирования

Устройства **CPE** должны резервировать энергонезависимое хранилище достаточной емкости для хранения **списков аннулирования**, согласно определениям в таблице B.2-1. **Доверительный орган ECI** следит за тем, чтобы выпущенные им списки аннулирования **ECI** соответствовали этим ограничениям.

Таблица B.2-1 – Максимальная длина списка аннулирования

Список аннулирования	Макс. число идентификаторов
Список аннулирования производителя	500
Список аннулирования хоста	500
Список аннулирования поставщика	500
Список аннулирования **клиента ECI**	500
Список аннулирования оператора	500
Список аннулирования **системы управления платформой**	500

B.3 Размер образа клиента ECI

Хост ECI должен иметь хранилище **образов клиента ECI** размером минимум 500 кбайт на каждый сегмент **клиента ECI**, который поддерживается этим хостом.

B.4 Параметры конфигурации карусели радиовещания

Интерфейс **ECI** определяет максимальные периоды обнаружения tCdownloadScenario для всех элементов, загружаемых из карусели радиовещания, с тем чтобы обеспечить надлежащую структуру **хостов ECI**. Параметр tCdownloadScenario отражает фактическое время загрузки, поэтому частота повторения карусели должна быть как минимум трехкратна этому параметру, что позволяет уложиться в указанные лимиты при загрузке **хостом ECI**. Радиовещательные организации должны обеспечивать полосу пропускания, достаточную для поддержки необходимой частоты повторения.

Интерфейс **ECI** также определяет максимальный размер модуля для распределения буфера.

Поле tCdownloadScenario и максимальный размер модуля, обработку которого должен поддерживать **хост ECI**, определены в таблице B.4-1.

Таблица B.4-1 – Максимальные периоды загрузки поля scenario и размеры модулей для каруселей ECI

Тип таблицы	tCdownloadScenario	Макс. размер модуля
Образы клиента ECI	5 минут	500 кбайт
Данные аннулирования **клиента ECI**	5 минут	100 кбайт на блок данных
Цепочка сертификатов системы управления платформой	10 секунд	50 кбайт
Данные аннулирования **системы управления платформой**	5 минут	100 кбайт на блок данных
Данные аннулирования **хоста ECI**	5 минут	100 кбайт на блок данных
Данные настройки **AS**	2 минуты	20 кбайт на блок данных

Приложение C

Обзор API хоста ECI

(Данное Приложение является неотъемлемой частью настоящей Рекомендации.)

В таблице C-1 приведены значения маркера **MsgApiTag**, описанного в пункте 9.3.1.

Таблица C-1 – Схема нумерации интерфейсов API ECI

API	Пункт	Значение MsgApiTag	Наивысшая версия API	Устаревшие версии API
Обнаружение интерфейса хоста	9.4.2	0x0001	0x0000	Нет
MMI	9.4.3	0x0002	0x0000	Нет
IP	9.4.4	0x0003	0x0000	Нет
HTTP(S)	9.4.4.6	0x0004	0x0000	Нет
Файловая система	9.4.5	0x0005	0x0000	Нет
Таймер и часы	9.4.6	0x0006	0x0000	Нет
Управление электропитанием	9.4.7	0x0007	0x0000	Нет
Языковые и страновые настройки	9.4.8	0x0008	0x0000	Нет
Усовершенствованная система безопасности (общий)	9.5.2.2	0x0009	0x0000	Нет
API дешифрования усовершенствованной системы безопасности	9.5.2.3	0x000A	0x0000	Нет
Экспорт усовершенствованной системы безопасности	9.5.2.4	0x000B	0x0000	Нет
Шифрование усовершенствованной системы безопасности	9.5.2.5	0x000C	0x0000	Нет
Смарт-карта	9.5.3	0x000D	0x0000	Нет
Карусель передачи данных	9.5.4	0x000E	0x0000	Нет
Дешифрование	9.6.2	0x000F	0x0000	Нет
Соединение экспорта	9.7.2.3	0x0010	0x0000	Нет
Соединение импорта	9.7.2.4	0x0011	0x0000	Нет
Повторное шифрование	9.7.2.5	0x0012	0x0000	Нет
Дешифрование микроклиента	9.7.2.6	0x0013	0x0000	Нет
Стандартная URI	9.8.2.3	0x0014	0x0000	Нет
Пользовательская URI	9.8.2.4	0x0015	0x0000	Нет
Базовая URI	9.8.2.5	0x0016	0x0000	Нет
Управление выходными данными	9.8.2.6	0x0017	0x0000	Нет
Защита водяными знаками	9.8.2.7	0x0018	0x0000	Нет
Родительский контроль	9.8.2.8	0x0019	0x0000	Нет
Синхронизация свойств контента	9.8.2.9	0x0020	0x0000	Нет
Родительская аутентификация	9.8.2.10	0x0021	0x0000	Нет
Делегирование родительской аутентификации	9.8.2.11	0x0022	0x0000	Нет
Связь между клиентами	9.9.2	0x0023	0x0000	Нет

Приложение D

Прямая совместимость определений свойств контента

(Данное Приложение является неотъемлемой частью настоящей Рекомендации.)

Свойства контента должны реализовываться максимально устойчивым способом с использованием аппаратного или низкоуровневого встроенного программного обеспечения. Изменение или обновление свойств контента после создания сообщений SOC может быть связано со сложностями, большими расходами или может быть невозможно. В настоящем разделе описывается подход к созданию путей развития для подобных свойств контента, который может быть реализован несмотря на указанные в данном разделе ограничения.

В будущем могут потребоваться новые свойства контента и/или расширение функций существующих свойств, что может включать увеличение количества битов, представляющих значение свойства контента. Реализация свойства контента в ранее созданных **хостах ECI** происходит без информации о новых функциях, поэтому его обновление часто не представляется возможным. Определение свойств контента в **хостах ECI** позволяет достичь максимальной прямой совместимости (с последующими версиями) по отношению к новым функциональным возможностям свойств контента.

Хосты ECI будут работать по алгоритму, определенному для всех входных значений, и игнорировать все расширения полей, для которых они не предназначены. Кроме того, они реализуют заданную логику работы, то есть каждому значению будущего свойства контента будет соответствовать *одна определенная логика работы* на всех **хостах ECI**, не реализующих все расширения, включая те **хосты ECI**, которые соответствуют первой версии свойства контента. При использовании этого принципа могут быть присвоены новые значения свойства контента и получена полная информация о действиях, которые могут привести к предыдущим версиям, реализуемых **хостом ECI**. Если новое свойство контента будет иметь два (или более) разных варианта для интерпретации обратной совместимости ранних версий **хостов ECI**, то могут быть присвоены два (или более) зарезервированных значения с той же семантикой новых свойств контента в определении новых свойств контента, но при этом каждое значение будет иметь надлежащую (но отличную от других) интерпретацию обратной совместимости.

Примером расширения поля является, например, новое поле управления выходными данными, которое должно быть определено для нового типа выходных данных X в API управления выходными данными. Это значение присваивается 5-му биту, который зарезервирован в версии 1. Может использоваться семантический эквивалент поля OcIP. Во всех предыдущих реализациях **клиентов ECI** этому полю присваивается значение 0. Интерпретация более поздней версией **хоста ECI** выглядит следующим образом:

- если OcAnyOther==0b0, OutputX разрешен;
- если OcAnyOther==0b1, OutputX не разрешен.

Это в точности соответствует семантике новой реализации **хоста ECI**, когда OcX == 0b0. Однако при OcX == 0b1 разрешение на вывод данных будет противоположным предшествующей конфигурации с OcX == 0b0, что позволит использовать новые функции в комбинации нового **хоста ECI** и нового **клиента ECI**. Следует отметить, что обратная интерпретация значений поля в зависимости от OcAnyOther гарантирует, что нулевое значение для любого неопределенного поля имеет присущий ему смысл: максимальное разрешение для OcAnyOther == 0b0 (вывод других данных разрешен) и минимальное разрешение для OcAnyOther = 0b1 (вывод других данных не разрешен).

И наоборот, важно, чтобы **клиенты ECI**, не использующие последнее определение свойств контента, не могли непреднамеренно обращаться к функциональным возможностям новых свойств контента из более поздних определений, которые им не известны, или, что еще хуже, не использовали подобные (предположительно, неприсвоенные) значения для решения частных задач на основе того факта, что такие значения изменяются по определенному алгоритму на всех **хостах ECI**. Такого рода ненадлежащее использование, как правило, создает серьезные препятствия для применения этих значений при решении определенных **ECI** задач. Таким образом, данная спецификация явным образом запрещает **клиентами ECI** применять неприсвоенные значения свойств контента.

В частности: для полей, которые могут иметь несколько значений, все зарезервированные значения будут изменяться по определенному алгоритму в **хостах ECI**, но зарезервированные значения не будут использоваться **клиентами ECI**.

Любое неприсвоенное субполе в определении свойства контента должно подчиняться сформулированному в **хосте ECI** определенному алгоритму, который соответствует одному из определенных значений свойства контента. Как правило, **хост ECI** игнорирует такие субполя, то есть **хост ECI** интерпретирует значение свойства контента применительно к полям, значение которых определено. В большинстве случаев **клиенты ECI** присваивают такому субполю значение 0. Любое отклонение политики неприсвоенного субполя, равного нулю, должно быть предопределено какой-либо версией определения свойства контента.

Все расширения поля должны игнорироваться **хостами ECI**, соответствующими определению свойства контента, а **клиенты ECI**, присваивающие значения, должны назначать таким расширениям полей значение 0.

Дополнение I

Список всех имеющихся сообщений API в алфавитном порядке

(Данное Дополнение не является неотъемлемой частью настоящей Рекомендации.)

Сообщения API, перечисленные в Дополнении I, взяты из следующих таблиц раздела 9 настоящей Рекомендации и перечислены в таблице I.1.

Таблица I.1 – Список таблиц, в которых приведены сообщения различных интерфейсов API

API	Таблица	Категория API
API обнаружения интерфейса хоста	9.4.2.1-1	Общие интерфейсы API
API пользовательского интерфейса	9.4.3.1-1	
API IP-сокета	9.4.4.3.1-1	
API UDP-сокета	9.4.4.4.1-1	
API TCP-сокета	9.4.4.5.1-1	
API HTTP Get	9.4.4.6.1-1	
API открытия/закрытия файлов	9.4.5.2.1-1	
API доступа к файлам	9.4.5.3.1-1	
API службы управления каталогом	9.4.5.4.1-1	
API таймера	9.4.6.2.1-1	
API часов	9.4.6.3.1-1	
API переключения режимов электропитания	9.4.7.2-1	
API выхода из режима ожидания	9.4.7.3-1	
API языковой и страновой настройки	9.4.8.1-1	
Усовершенствованная система безопасности (общий API)	9.5.2.2.1-1	API с поддержкой **ECI**
API дешифрования усовершенствованной системы безопасности	9.5.2.3.1-1	
API экспорта усовершенствованной системы безопасности	9.5.2.4.1-1	
API шифрования усовершенствованной системы безопасности	9.5.2.5.1-1	
API управления сеансом **смарт-карты**	9.5.3.6.1-1	
API связи со **смарт-картой**	9.5.3.6.1-1	
API обнаружения карусели данных	9.5.4.1-1	
API сеанса дешифрования указателя медиаданных	9.6.2.2.1-1	
API соединения экспорта	9.7.2.3.1-1	
API соединения импорта	9.7.2.4.1-1	
API повторного шифрования	9.7.2.5.1-1	
API дешифрования	9.7.2.6.1-1	
API прав на использование и родительского контроля	9.8.2.1-1	
API связи между клиентами	9.9.2.1-1	

В таблице I.2 приведен список всех сообщений API в алфавитном порядке.

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
1	callAsNextKeySession	Усовершенствованная система безопасности (общая)	9.5.2.2.3	S	C→H	Переход к следующему случайному ключу для сеанса
2	callCardGetProp	**Смарт-карта**	9.5.3.6.5	S	H→C	Получение свойства/параметра связи карты
3	callCardSessionPrio	**Смарт-карта**	9.5.3.5.3	S	C→H	Задание приоритета сеанса **смарт-карты**
4	callCardSetProp	**Смарт-карта**	9.5.3.6.4	S	H→C	Установка параметра связи карты
5	callFileDataLog	Файловая система	9.4.5.3.6	S	C→H	Добавляет данные в конце буферизованного файла
6	callLocaltime	Часы	9.4.6.3.3	S	C→H	Преобразует целочисленное значение времени в местное время
7	getApis	Обнаружение интерфейса	9.4.2.2	S	C→H	Получение доступных интерфейсов API хоста
8	getApiVersions	Обнаружение интерфейса	9.4.2.3	S	C→H	Получение доступных версий API хоста
9	getAsClientRnd	Усовершенствованная система безопасности (общая)	9.5.2.2.13	S	C→H	Получение нового случайного числа для приложений **клиента ECI**
10	getAsSC	Усовершенствованная система безопасности (общая)	9.5.2.2.14	S	C→H	Получение текущего статуса поля управления скремблированием контента в сеансе
11	getAsSessionLimitCounter	Усовершенствованная система безопасности (общая)	9.5.2.2.10	S	C→H	Получение текущего предельного значения счетчика для сеанса
12	getAsSessionRk	Усовершенствованная система безопасности (общая)	9.5.2.2.9	S	C→H	Получение случайного значения ключа для сеанса
13	getAsSlotRk	Усовершенствованная система безопасности (общая)	9.5.2.2.8	S	C→H	Получение случайного значения ключа для **сегмента AS**
14	getCardConnStatus	**Смарт-карта**	9.5.3.5.4	S	H→C	Предоставление статуса состояния подключения к карте
15	getChipsetId	Усовершенствованная система безопасности (общая)	9.5.2.2.16	S	C→H	Получение значения параметра ChipsetID **блока лестницы ключей**
16	getDcrMarkMeta	Свойство контента	9.8.2.7.4	S	H→C	Считывание свойства системы маркирования
17	getDcrMarkSyst	Свойство контента	9.8.2.7.2	S	H→C	Получение поддерживаемых систем маркирования
18	getDcrTsSource	Управление источником TS дешифрования	9.6.2.3.6.2	S	C→H	**Клиент ECI** получает источник транспортного потока
19	getEncrStdUri	Свойство контента	9.8.2.3.2	S	C→H	Получение стандартной URI для контента, подлежащего повторному шифрованию
20	getEncrBasicUri	Свойство контента	9.8.2.5.2	S	C→H	Получение базовой URI для контента, подлежащего повторному шифрованию
21	getEncrCustUri	Свойство контента	9.8.2.4.2	S	C→H	Получение пользовательской URI для контента, подлежащего повторному шифрованию
22	getEncrOutputCtrl	Свойство контента	9.8.2.6.2	S	C→H	Получение ограничений по управлению выходными данными для контента, подлежащего повторному шифрованию
23	getEncrParCtrl	Свойство контента	9.8.2.8.2	S	C→H	Получение условий родительского контроля для контента, подлежащего дескремблированию

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
24	getIccClientInfo	Связь между клиентами	9.9.2.4	S	C→H	**Клиент ECI** считывает идентификационные данные и статус соединения другого **клиента ECI** в системе
25	getIccMaxClients	Связь между клиентами	9.9.2.2	S	C→H	**Клиент ECI** считывает максимальное число **клиентов ECI**, которое может поддерживать **хост ECI**
26	getImageTargetId	Усовершенствованная система безопасности (общая)	9.5.2.2.17	S	C→H	Получение значения параметра ECI_Image_Target_Id оборудования CPE
27	getPwrStatus	Управление электропитанием	9.4.7.2.2	S	C→H	Получает сообщение о текущем состоянии электропитания
28	getTime	Часы	9.4.6.3.2	S	C→H	Считывает показания локальных системных часов в виде целого числа
29	reqAsAStartDecryptSession	API дешифрования усовершенствованной системы безопасности	9.5.2.3.2	A	C→H	Запуск сеанса дешифрования в **сегменте AS клиента ECI**
30	reqAsAuthDecrSlotConfig	API дешифрования усовершенствованной системы безопасности	9.5.2.3.4	A	H→C	Аутентификация конфигурации сегмента с применением механизмов аутентификации (режим дешифрования)
31	reqAsAuthEncrSlotConfig	Шифрование усовершенствованной системы безопасности	9.5.2.5.5	A	C→H	Аутентификация конфигурации сегмента и параметров шифрования с применением механизмов аутентификации (режим шифрования)
32	reqAsClientChalResp	Усовершенствованная система безопасности (общая)	9.5.2.2.7	A	C→H	Применение ключа аутентификации **клиента ECI** к данным и возвращаемому результату
33	reqAsComputeAkClient	Усовершенствованная система безопасности (общая)	9.5.2.2.6	A	C→H	Вычисление ключа аутентификации для приложений **клиента ECI**
34	reqAsComputeEncrCw	Шифрование усовершенствованной системы безопасности	9.5.2.5.4	A	C→H	Вычисление слова управления шифрования
35	reqAsEventCpChange	Шифрование усовершенствованной системы безопасности	9.5.2.5.8	A	H→C	Сообщение о событии изменения свойств импортированного контента во время сеанса шифрования
36	reqAsEventSC	Усовершенствованная система безопасности (общая)	9.5.2.2.15	A	H→C	Сообщение о событии изменения поля управления скремблированием в сеансе
37	reqAsEventSessionLimit	Усовершенствованная система безопасности (общая)	9.5.2.2.12	A	H→C	Отправка события **клиенту ECI** по достижении предельного значения для оставшихся единиц
38	reqAsExportConnEnd	Экспорт усовершенствованной системы безопасности	9.5.2.4.3	A	C→H	Завершает текущий сеанс экспорта
39	reqAsExportConnSetup	Экспорт усовершенствованной системы безопасности	9.5.2.4.2	A	C→H	Настройка **соединения экспорта** от сеанса дешифрования до сеанса шифрования
40	reqAsInitSlot	Усовершенствованная система безопасности (общая)	9.5.2.2.2	A	C→H	Инициализирует **сегмент AS**
41	reqAsLdUssk	Шифрование усовершенствованной системы безопасности	9.5.2.5.6	A	C→H	Загрузка секретного ключа **микросервера**
42	reqAsLoadSlotLk	Усовершенствованная система безопасности (общая)	9.5.2.2.5	A	C→H	Вычисление ключа связи верхнего уровня (LK1)

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
43	reqAsMInikLk1	Шифрование усовершенствованной системы безопасности	9.5.2.5.7	A	C→H	Вычисление асимметричного сообщения инициализации **микроклиента**
44	reqAsStartEncryptSession	Шифрование усовершенствованной системы безопасности	9.5.2.5.3	A	C→H	Запуск сеанса шифрования
45	reqAsStopSession	Усовершенствованная система безопасности (общая)	9.5.2.2.4	A	C→H	Остановка сеанса
46	reqCardCmdRes	**Смарт-карта**	9.5.3.6.2	A	C→H	Отправка команды карте, получение отклика карты
47	reqCardReInit	**Смарт-карта**	9.5.3.6.3	A	C→H	Перезагрузка карты ("теплая" или "холодная") и повтор последовательности инициализации с приоритетом последней настройки инициализации
48	reqCCardConClose	**Смарт-карта**	9.5.3.5.6	A	H→C	Сообщает **клиенту ECI**, что сеанс работы с картой закрыт
49	reqCCardConOpen	**Смарт-карта**	9.5.3.5.5	A	H→C	Сообщает **клиенту ECI**, что сеанс работы с картой открыт
50	reqCCountry	Страна	9.4.8.2.2	A	H→C	**Хост ECI** запрашивает актуальную информацию о предпочитаемой стране **клиента ECI**
51	reqCLanguage	Язык	9.4.8.2.4	A	H→C	**Хост ECI** запрашивает актуальную информацию о предпочитаемом языке **клиента ECI**
52	reqCpChange	Свойство контента	9.8.2.9.2	A	H→C	**Хост ECI** сигнализирует о предстоящем изменении свойств контента, подлежащего повторному шифрованию
53	reqDCAcqModule	Получение карусели данных	9.5.4.3	A	C→H	**Клиент ECI** посылает запрос **хосту ECI** на получение определенного модуля карусели данных **ECI** в файл с применением параметров фильтра модуля и различных режимов
54	reqDCAcqGroupInfo	Получение карусели данных	9.5.4.2	A	C→H	**Клиент ECI** посылает запрос **хосту ECI** на считывание структуры GroupInfoIndication в DSI-сообщении указанной карусели данных **ECI**
55	reqDcrFileQuit	Медиафайл дешифрования	9.6.2.4.4.4	A	C→H	**Клиент ECI** отменяет сеанс дескремблирования с **хостом ECI**
56	reqDcrFileData	Запрос данных через файловый фильтр	9.6.2.4.5.2.4	A	C→H	**Клиент ECI** направляет **хосту ECI** запрос на сбор данных через фильтр файлов
57	reqDcrFileStop	Медиафайл дешифрования	9.6.2.4.4.3	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на остановку операции дескремблирования сеанса **указателя медиаданных**
58	reqDcrFileFilter	Запрос фильтра файлов	9.6.2.4.5.2.3	A	C→H	**Клиент ECI** направляет **хосту ECI** запрос на установку фильтра данных для сбора данных по безопасности
59	reqDcrFileKeyComp	Запрос на вычисление ключа	9.6.2.4.6.3	A	H→C	Инициализация требуемого вычисления или других действий **клиента ECI** для формирования слова управления при доступном идентификаторе ключа
60	reqDcrFileStart	Медиафайл дешифрования	9.6.2.4.4.2	A	H→C	Посылает **клиенту ECI запрос** на дескремблирование или возврат статуса дескремблирования файла или потока
61	reqDcrIpServer	Повторное шифрование	9.7.2.6.5	A	C→H	**Микроклиент** направляет **хосту ECI** запрос на предоставление IP-адреса **микросервера** для дополнительной связи, относящейся к сеансу **указателя медиаданных**

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
62	reqDcrMhBcAlloc	Дешифрование указателя медиаданных	9.6.2.2.5	A	C→H	**Клиент ECI** запрашивает сеанс **указателя медиаданных** для собственной радиовещательной сети доступа
63	reqDcrMhCancel	Дешифрование указателя медиаданных	9.6.2.2.6	A	C→H	**Клиент ECI** отменяет медиасеанс с **хостом ECI**
64	reqDcrMhClose	Дешифрование указателя медиаданных	9.6.2.2.4	A	H→C	**Хост ECI** закрывает медиасеанс с **клиентом ECI**
65	reqDcrMhOpen	Дешифрование указателя медиаданных	9.6.2.2.3	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на открытие специального медиасеанса с использованием **указателя медиаданных**
66	reqDcrMsgRecv	Повторное шифрование	9.7.2.6.7	A	H→C	**Хост ECI** передает **микроклиенту** сообщение от **микросервера** относительно сеанса **указателя медиаданных**
67	reqDcrMsgSend	Повторное шифрование	9.7.2.6.6	A	C→H	**Микроклиент** направляет **хосту ECI** запрос на отправку сообщения **микросерверу** относительно сеанса **указателя медиаданных**
68	reqDcrTargetCred	Повторное шифрование	9.7.2.6.4	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на предоставление данных инициализации для соединения **микросервера**, обычно используемого для аутентификации целевого объекта
69	reqDcrTargets	Повторное шифрование	9.7.2.6.3	A	H→C	**Хост ECI** направляет **микроклиенту** запрос на предоставление целевых объектов шифрования, которым он может оказать услуги дешифрования
70	reqDcrTsData	Повторное шифрование	9.7.2.6.8	A	C→H	**Микросервер** предоставляет **хосту ECI** данные для переадресации на **целевой микроклиент указателя медианных** для дешифрования, включая информацию синхронизации для сообщений ECM
71	reqDcrTsDescrquit	Дешифрование контента транспортного потока	9.6.2.3.4.4	A	C→H	**Клиент ECI** направляет **хосту ECI** запрос на прекращение дескремблирования **сеанса указателя медиаданных**
72	reqDcrTsData	Повторное шифрование микроклиента	6.7.2.6.7	A	H→C	**Хост ECI** передает микроклиенту данные, необходимые **указателю медиаданных** для дешифрования контента в (ближайшем) будущем
73	reqDcrTsDescrStop	Дешифрование контента транспортного потока	9.6.2.3.4.3	A	H→C	**Хост ECI** посылает **клиенту ECI** запрос на остановку дескремблирования **сеанса указателя медиаданных**
75	reqDcrTsDescrStart	Дешифрование контента транспортного потока	9.6.2.3.4.2	A	H→C	Запрос **клиенту ECI** на дескремблирование или возврат статуса дескремблирования программы в транспортном потоке
76	reqDcrTsRelocate	Управление источником TS дешифрования	9.6.2.3.6.3	A	C→H	**Клиент ECI** перемещает источник транспортного потока
77	reqDcrTsSection	Сбор данных TS дешифрования	9.6.2.3.5.5	A	H→C	Переадресует обнаруженную секцию **клиенту ECI**
78	reqDcrTsSelectCancel	Управление источником TS дешифрования	9.6.2.3.6.6	A	C→H	**Клиент ECI** отменяет выбранную ранее программу

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
79	reqDcrTsSelectPmt	Управление источником TS дешифрования	9.6.2.3.6.5	A	C→H	**Клиент ECI** выбирает программу в транспортном потоке по PMT
80	reqDcrTsSelectPrg	Управление источником TS дешифрования	9.6.2.3.6.4	A	C→H	**Клиент ECI** выбирает по номеру программу в транспортном потоке
81	reqDcrTsTable	Сбор данных TS дешифрования	9.6.2.3.5.6	A	C→H	**Клиент ECI** обнаруживает таблицу в потоке
82	reqEncrConnDrop	Повторное шифрование	9.7.2.5.5	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на исключение любой информации по предварительно аутентифицированному соединению повторного шифрования
83	reqEncrConnSetup	Повторное шифрование	9.7.2.5.4	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на создание **соединения целевого объекта** повторного шифрования и предварительную аутентификацию **целевого объекта** повторного шифрования для последующей ссылки при настройке сеанса **указателя медиаданных**
84	reqEncrFileData	Повторное шифрование	9.7.2.5.18	A	C→H	**Микросервер** предоставляет **хосту ECI** сообщение для переадресации на **целевой микроклиент указателя медианных** для дешифрования, включая информацию синхронизации для KeyID
85	reqEncrIpServer	Повторное шифрование	9.7.2.5.13	A	H→C	**Хост ECI** запрашивает адрес IP-сервера **микросервера** для того, чтобы разрешить **микроклиентам** создавать IP-соединения
86	reqEncrMhCancel	Повторное шифрование	9.7.2.5.9	A	C→H	**Клиент ECI** завершает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт
87	reqEncrMhClose	Повторное шифрование	9.7.2.5.8	A	H→C	**Хост ECI** закрывает сеанс повторного шифрования с **клиентом ECI**
88	reqEncrMhOpen	Повторное шифрование	9.7.2.5.7	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на открытие сеанса **указателя медиаданных** для повторного шифрования контента из входящего **соединения импорта** для установленного соединения повторного шифрования
89	reqEncrMhQuit	Повторное шифрование	9.7.2.5.12	A	C→H	**Клиент ECI** информирует **хост ECI** о том, что операция повторного шифрования **указателя медиаданных** прекращена
90	reqEncrMhStart	Повторное шифрование	9.7.2.5.10	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на запуск операции повторного шифрования для сеанса **указателя медиаданных**
91	reqEncrMhStop	Повторное шифрование	9.7.2.5.11	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на остановку операции повторного шифрования для сеанса **указателя медиаданных**
92	reqEncrMsgRecv	Повторное шифрование	9.7.2.5.18	A	H→C	**Хост ECI** передает **микросерверу** сообщение от **целевого объекта** относительно сеанса **указателя медиаданных**
93	reqEncrMsgSend	Повторное шифрование	9.7.2.5.14	A	C→H	**Микросервер** направляет **хосту ECI** запрос на переадресацию сообщения **целевому объекту** относительно сеанса **указателя медиаданных**

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
94	reqEncrTargets	Повторное шифрование	9.7.2.5.3	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на предоставление узлов **целевого объекта**, которые он может аутентифицировать
95	reqEncrTsData	Повторное шифрование	9.7.2.5.16	A	C→H	**Микросервер** предоставляет **хосту ECI** данные для переадресации на **целевой микроклиент указателя медианных** для дешифрования, включая информацию синхронизации для сообщений ECM
96	reqEncrTsEcm	Повторное шифрование	9.7.2.5.17	A	C→H	**Микросервер** предоставляет секцию ECM, которую **микроклиент** запрашивает для дешифрования в следующем криптопериоде
97	reqExpConnCancel	Соединение экспорта	9.7.2.3.5	A	C→H	**Клиент ECI** завершает инициализированное **соединение экспорта** с **клиентом ECI**, выполняющим импорт
98	reqExpConnDrop	Соединение экспорта	9.7.2.3.4	A	H→C	**Хосты ECI** отменяют любое предварительно инициализированное соединение **клиента ECI**, выполняющего экспорт, с **клиентом ECI**, выполняющим импорт
99	reqExpConnNodes	Соединение экспорта	9.7.2.3.2	A	H→C	**Хост ECI** запрашивает у **клиента ECI** дополнительные узлы экспорта
100	reqExpConnSetup	Соединение экспорта	9.7.2.3.3	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на инициализацию **соединения экспорта** с **клиентом ECI**, выполняющим импорт, на основе **цепочки импорта**
101	reqExpMhCancel	Соединение экспорта	9.7.2.3.8	A	C→H	**Клиент ECI** отменяет сеанс экспорта
102	reqExpMhClose	Соединение экспорта	9.7.2.3.7	A	H→C	**Хост ECI** закрывает сеанс экспорта
103	reqExpMhOpen	Соединение экспорта	9.7.2.3.6	A	H→C	**Хост ECI** направляет **клиенту ECI** запрос на создание сеанса экспорта на основе ранее инициализированного **соединения экспорта**
104	reqFileClose	Файловая система	9.4.5.2.3	A	C→H	Закрывает открытый файл
105	reqFileCreate	Файловая система	9.4.5.4.3	A	C→H	Создание нового файла
106	reqFileDelete	Файловая система	9.4.5.4.4	A	C→H	Удаление файла
107	reqFileDir	Файловая система	9.4.5.4.5	A	C→H	Перечисляет имена файлов, доступных в файловой системе **клиентов ECI**
108	reqFileOpen	Файловая система	9.4.5.2.2	A	C→H	Открывает частный файл **клиента ECI**
109	reqFileRead	Файловая система	9.4.5.3.3	A	C→H	Считывает последовательные байты, начиная с текущего расположения файла
110	reqFileRemoveData	Файловая система	9.4.5.3.5	A	C→H	Удаляет данные из файла в текущем расположении
111	reqFileSeek	Файловая система	9.4.5.3.4	A	C→H	Перепозиционирует текущее расположение файла
112	reqFileStat	Файловая система	9.4.5.4.2	A	C→H	Возврат размера и времени изменения файла
113	reqFileWrite	Файловая система	9.4.5.3.2	A	C→H	Записывает последовательные байты, начиная с текущего расположения файла
114	reqHCardConClose	**Смарт-карта**	9.5.3.5.7	A	C→H	Сообщает **хосту ECI**, что **клиент ECI** намерен завершить сеанс работы с подключенной картой
115	reqHCountry	Страна	9.4.8.2.1	A	C→H	Запрашивает актуальную информацию о предпочитаемой стране **хоста ECI**

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
116	reqHLanguage	Язык	9.4.8.2.3	A	C→H	Запрашивает актуальную информацию о предпочитаемом языке **хоста ECI**
117	reqHttpGetData	HTTP Get	9.4.4.6.3	A	C→H	Выполняет запрос HTTP Get по URL и передает результат в виде данных клиенту
118	reqHttpGetFile	HTTP Get	9.4.4.6.3	A	C→H	Выполняет запрос HTTP Get по URL и сохраняет результат в файле
119	reqIccPipeCancel	Связь между клиентами	9.9.2.7	A	C→H	**Клиент ECI** отменяет канал связи
120	reqIccPipeClose	Связь между клиентами	9.9.2.8	A	H→C	**Хост ECI** информирует **клиент ECI**, что канал связи закрыт
121	reqIccPipeMsgRecv	Связь между клиентами	9.9.2.10	A	H→C	**Клиент ECI** получает сообщение от своего партнера по каналу связи
122	reqIccPipeMsgSend	Связь между клиентами	9.9.2.9	A	C→H	**Клиент ECI** отправляет сообщение своему партнеру по каналу связи
123	reqIccPipeOpen	Связь между клиентами	9.9.2.5	A	C→H	Запрос на открытие канала связи с другим **клиентом ECI**
124	reqIccPipeOpenReq	Связь между клиентами	9.9.2.6	A	H→C	Входящий запрос от другого **клиента ECI** на открытие канала связи
125	reqIccSystemReady	Связь между клиентами	9.9.2.3	A	H→C	**Хост ECI** информирует **клиент ECI**, что все **клиенты ECI** инициализированы
126	reqImpConnCancel	Соединение импорта	9.7.2.4.6	A	C→H	**Клиент ECI** завершает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт
127	reqImpConnChain	Соединение импорта	9.7.2.4.3	A	H→C	**Хост ECI** направляет **клиенту ECI,** выполняющему импорт, запрос на предоставление цепочки ввода данных для конкретного узла импорта
128	reqImpConnChainRenew	Соединение импорта	9.7.2.4.3	A	C→H	**Клиент ECI** направляет **хосту ECI** запрос на повторную инициализацию соединения с использованием обновленной **цепочки импорта**
129	reqImpConnDrop	Соединение импорта	9.7.2.4.5	A	H→C	**Хост ECI** прекращает **соединение импорта** с заданным **клиентом ECI**, выполняющим экспорт
130	reqImpConnNodes	Соединение импорта	9.7.2.4.2	A	H→C	**Хост ECI** направляет **клиенту ECI**, выполняющему импорт, запрос на предоставление его узлов импорта
131	reqImpConnSetup	Соединение импорта	9.7.2.4.4	A	H→C	**Хост ECI** направляет **клиенту ECI**, выполняющему импорт, запрос на инициализацию соединения импорта с конкретным **клиентом ECI**, выполняющим экспорт, через узел импорта
132	reqIpAddrinfo	IP-сокеты	9.4.4.3.4	A	C→H	Получает адрес (удаленного) **хоста ECI**
133	reqIpClose	IP-сокеты	9.4.4.3.3	A	C→H	Закрывает IP-сокет **ECI**
134	reqIpSocket	IP-сокеты	9.4.4.3.2	A	C→H	Открывает IP-сокет **ECI**
135	reqIpTcpAccept	Сокет TCP/IP	9.4.4.5.5	A	C→H	Одноранговый узел сервера TCP принимает подключение от однорангового узла клиента TPC
136	reqIpTcpConnect	Сокет TCP/IP	9.4.4.5.2	A	C→H	Клиент TCP подключается к одноранговому узлу сервера TCP
137	reqIpTcpRecv	Сокет TCP/IP	9.4.4.5.4	A	C→H	Получает данные от подключенного однорангового узла
138	reqIpTcpSend	Сокет TCP/IP	9.4.4.5.3	A	C→H	Направляет данные подключенному одноранговому узлу
139	reqIpUdpRecvMsg	Сокет UDP/IP	9.4.4.4.3	A	C→H	Принимает сообщение от однорангового порта UDP

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
140	reqIpUdpSendMsg	Сокет UDP/IP	9.4.4.4.2	A	C→H	Отправляет сообщение на одноранговый порт UDP
141	reqParAuthChk	Свойство контента	9.8.2.10.3	A	C→H	Запрос **хосту ECI** на проведение родительской аутентификации от имени **клиента ECI**
142	reqParAuthChkCan	Свойство контента	9.8.2.10.4	A	C→H	Отменяет предыдущий запрос на родительскую аутентификацию, направленный хосту
143	reqParAuthCid	Свойство контента	9.8.2.10.5	A	H→C	Запрашивает PIN-код для родительской авторизации для (будущего) элемента контента, подлежащего декодированию. Таким образом может быть запущен диалог родительской аутентификации
144	reqParAuthDel	Свойство контента	9.8.2.11.2	A	H→C	**Хост ECI** делегирует родительскую аутентификацию **клиенту ECI**
145	reqParAuthDelCan	Свойство контента	9.8.2.11.3	A	H→C	**Хост ECI** отменяет предыдущий запрос на родительскую аутентификацию, направленный **клиенту ECI**
146	reqPwrChange	Управление электропитанием	9.4.7.2.4	A	H→C	Оповещение об изменении состояния электропитания
147	reqTimerCancel	Таймер	9.4.6.2.3	A	C→H	Отменяет ранее установленное событие таймера
148	reqTimerEvent	Таймер	9.4.6.2.2	A	C→H	Устанавливает событие таймера в будущем
149	reqUiClientQuery	Пользовательский интерфейс	9.4.3.4.8	A	H→C	**Клиент ECI** принимает запрос от HTML-приложения в браузере и отправляет (динамический) отклик
150	reqUiContainerMount	Пользовательский интерфейс	9.4.3.4.2	A	C→H	Монтирует контейнер приложения пользовательского интерфейса с ресурсами HTML для поддержки сеансов через пользовательский интерфейс
151	reqUiSessionCancel	Пользовательский интерфейс	9.4.3.4.7	A	H→C	**Хост ECI** отменяет сеанс через **пользовательский** интерфейс
152	reqUiSessionClose	Пользовательский интерфейс	9.4.3.4.6	A	C→H	**Клиент ECI** завершает сеанс через **пользовательский** интерфейс
153	reqUiSessionCommence	Пользовательский интерфейс	9.4.3.4.4	A	H→C	**Хост ECI** предлагает **клиенту ECI** открыть сеанс через пользовательский интерфейс
154	reqUiSessionOpen	Пользовательский интерфейс	9.4.3.4.5	A	C→H	Клиент **ECI** отправляет запрос на открытие сеанса связи с **пользователем** через **пользовательский** интерфейс и отображение контента на экране
155	reqPwrWakeupEvent	Управление электропитанием	9.4.7.3	A	H→C	Сигнализирует об истечения времени таймера выхода из режима ожидания
156	setApiVersion	Обнаружение интерфейса	9.4.2.4	S	C→H	Задание используемой версии API хоста
157	setAsPermitCPChange	Шифрование усовершенствованной системы безопасности	9.5.2.4	S	C→H	Включение и отключение изменений свойств импортированного контента путем выбора слова управления для шифрования во время сеанса шифрования
158	setAsSC	Шифрование усовершенствованной системы безопасности	9.5.2.4	S	C→H	Установка поля управления скремблированием для зашифрованного контента во время сеанса шифрования
159	setAsSessionLimitEvent	Усовершенствованная система безопасности (общий)	9.5.2.5.11	S	C→H	Установка предельного значения для отправки сообщения reqAsEventSessionLimit **клиенту ECI**

Таблица I.2 – Список всех сообщений API в алфавитном порядке

№	Сообщение	API	Пункт	Тип	Направление	Описание
160	setCardMatch	**Смарт-карта**	9.5.3.5.2	S	C→H	Задание списка спецификатора идентификации карты для **клиента ECI**
161	setCpSync	Свойство контента	9.8.2	S	C→H	**Клиент ECI** сигнализирует о том, что текущий набор свойств контента согласован и может применяться к контенту, подлежащему дескремблированию с использованием следующего слова управления
162	setDcrBasicUri	Свойство контента	9.8.2.5.1	S	C→H	Установка базовой URI для контента, подлежащего дескремблированию
163	setDcrCustUri	Свойство контента	9.8.2.4.1	S	C→H	Установка пользовательской URI для контента, подлежащего дескремблированию
164	setDcrMarkBasic	Свойство контента	9.8.2.7.5	S	C→H	Установка базовых полезных данных маркирования для контента, подлежащего дескремблированию
165	setDcrMarkExt	Свойство контента	9.8.2.7.6	S	C→H	Установка расширенных полезных данных маркирования для контента, подлежащего дескремблированию
166	setDcrMarkMeta	Защита водяными знаками	9.8.2.7.3	S	C→H	Установка контрольного значения системы маркирования
167	setDcrMhMatch	Дешифрование указателя медиаданных	9.6.2.2.2	S	C→H	Сообщает **хосту ECI**, по каким идентификаторам может быть опознан **клиент ECI** для дескремблирования контента
168	setDcrModes	Повторное шифрование	9.7.2.6.1	S	C→H	**Микроклиент** информирует **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации)
170	setDcrOutputCtl	Свойство контента	9.8.2.6.1	S	C→H	Установка ограничений по управлению выходными данными для контента, подлежащего дескремблированию
171	setDcrParCtl	Свойство контента	9.8.2.8.1	S	C→H	Установка условий родительского контроля для контента, подлежащего дескремблированию
172	setDcrStdUri	Свойство контента	9.8.2.8.1	S	C→H	Установка стандартной URI для контента, подлежащего дескремблированию
173	setDcrTsSectionAcq	Сбор данных TS дешифрования	9.6.2.3.5.4	S	C→H	Задает фильтр для обнаружения секций
176	setDcrTsSectionAcqDefault	Сбор данных TS дешифрования	9.6.2.3.5.3	S	C→H	Задает фильтр по умолчанию для обнаружения секций
177	setEncrModes	Повторное шифрование	9.7.2.5.2	S	C→H	**Микросервер** информирует **хост ECI** о режимах, которые он поддерживает (режимы шифрования, режимы формата данных и режимы синхронизации)
178	setPwrInfo	Управление электропитанием	9.4.7.2.3	S	C→H	Запрашивает уведомление о событии в отношении изменения состояния электропитания
179	setUiClientAttention	Пользовательский интерфейс	9.4.3.4.3	S	C→H	**Клиент ECI** сообщает о намерении начать сеанс через пользовательский интерфейс, не связанный с **указателем медиаданных**
180	setPwrWakeup	Управление электропитанием	9.4.7.3	S	C→H	Устанавливает время выхода из режима ожидания для **клиента ECI**

Дополнение II

Тематические области, требующие доработки

(Данное Дополнение не является неотъемлемой частью настоящей Рекомендации.)

Определено, что настоящая Рекомендация нуждается в доработке и валидации, с тем чтобы обеспечить ее соответствие требованиям [ITU-T J.1010], и что необходимо обновить Рекомендацию [ITU-T J.1010], отразив в ней требования спецификации системы расширенной защиты контента MovieLabs (ECP) [b-ECP]. Рекомендации [ITU-T J.1011], ITU-T J.1012, [ITU-T J.1013], [ITU-T J.1014], [ITU-T J.1015] и [b-ITU-T J.1015.1] следует в дальнейшем обновить, отразив в них указанные изменения в [ITU-T J.1010].

Ряд Государств – Членов МСЭ наряду с заинтересованными сторонами со всего мира, представляющими самые разные отрасли, включая производителей устройств и электронных компонентов, владельцев и лицензиатов авторских прав на контент, поставщиков услуг на базе технологии over-the-top (OTT) и линейного телевидения, а также поставщиков решений для систем условного доступа (CAS) и управления цифровыми правами (DRM), выразили обеспокоенность тем, что встроенный общий интерфейс (ECI) не в полной мере отвечает требованиям ECP и требованиям к защите контента, предъявляемым в более широком круге отраслей.

Соответствующие вопросы были, в частности, подняты во вкладах к собранию 9-й Исследовательской комиссии МСЭ-Т (ИК9) (16–23 апреля 2020 года). Во вкладах, представленных Израилем, Австралией, Членом Сектора МСЭ-Т компанией Samsung, а также Ассоциированными членами ИК9 компаниями Sky Group и MovieLabs, предлагалось внести ряд изменений в Рекомендации по тематике ECI, однако согласие по ним достигнуто не было. Эти предложения перечислены в [b-SG9 Report 17 Ann.1].

Они состояли, в частности, в следующем:

1) упростить систему ECI, ограничив сферу ее применения;
2) отказаться от DRM;
3) отказаться от повторного шифрования контента;
4) отказаться от управления программным обеспечением;
5) добавить интерфейсы API для защищенного хранения и криптографических операций;
6) предусмотреть возможность использования лестниц ключей, определяемых поставщиком;
7) установить требования к TEE, изложенные в Рекомендации J.1207;
8) включить в Рекомендации реализацию TEE для виртуальной машины;
9) применять более стойкие алгоритмы шифрования, например SHA-384;
10) использовать стандартные сертификаты, подобные приведенным в Рекомендации МСЭ-Т X.509;
11) пересмотреть обмен данными между клиентами;
12) организовать дополнительное взаимодействие с ETCИ;
13) провести дополнительное коллегиальное рассмотрение;
14) рассмотреть возможные альтернативы модели доверительного органа;
15) уточнить технические аспекты правил соответствия и обеспечения устойчивости ECI;
16) добавить требования об обеспечении технического разнообразия, например о рандомизации распределения адресного пространства;
17) добавить требования о проверке целостности данных на этапе выполнения.

Эти предложения отражают постоянную эволюцию защиты контента и способов ее нарушения. Первоначальный замысел ECI возник почти за десять лет до утверждения настоящей Рекомендации МСЭ-Т. Системы, подобные ECI, необходимо регулярно оценивать на предмет стойкости к современным методам осуществления атак, а также соответствия отраслевым требованиям к защите.

Существуют и другие механизмы обеспечения функциональной совместимости. В частности, что касается применения DRM, большинство интернет-служб доставки видео внедрили другие решения, обеспечивающие функциональную совместимость наряду с решением стоящих перед этими службами задач.

Важной задачей является внесение большей ясности, так как многие Государства-Члены рассматривают стандарты МСЭ как авторитетные источники руководящих указаний по развитию их рынков и отраслей. Упомянутый выше список предложений призван способствовать тому, чтобы можно было в полной мере уяснить все последствия, связанные с настоящей Рекомендацией МСЭ-Т, при внедрении ECI на рынках этих государств и учесть все возможные вопросы при рассмотрении законодательных и нормативных актов и потребностей рынка, требующих обеспечения функциональной совместимости потребительского цифрового телевизионного оборудования. Он также позволяет производителям оборудования, предпочитающим брать за основу при проектировании особые наборы требований или иные стандарты, учитывать эти вопросы в процессе разработки продукции, предназначенной для различных рынков.

Библиография

[b-ITU-T J.1015.1] Recommendation ITU-T J.1015.1 (2020), *Embedded common interface for exchangeable CA/DRM solutions; Advanced security system – Key ladder block: Authentication of control word-usage rules information and associated data 1*.

[b-ITU-T J Suppl. 7] Supplement 7 to the ITU-T J series Recommendation(2020), *Embedded Common Interface for exchangeable CA/DRM solutions; Guidelines for the implementation of ECI.*

[b-SG9 Report 17 Ann.1] ITU-T SG9 meeting report, SG9-R17-Annex 1 (2020), Annex 1 to Report 17 of the SG9 fully virtual meeting held 16-23 April 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 800-52r2] NIST SP 800-52 rev2 (August 2019): "Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations".

[b-CI Plus] CI Plus Specification V1.3.1 (2011-09).
Доступно по адресу 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*" revision 2.3., Feb 28, 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, No. 3, August 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 – Version 1.2. https://movielabs.com/ngvideo/MovieLabs_ECP_Spec_v1.2.pdf

Все гиперссылки, включенные в данный раздел, были действительны на момент публикации, однако их действительность в долгосрочной перспективе не может быть гарантирована.

СЕРИИ РЕКОМЕНДАЦИЙ МСЭ-Т

Серия A	Организация работы МСЭ-Т
Серия D	Принципы тарификации и учета и экономические и стратегические вопросы международной электросвязи/ИКТ
Серия E	Общая эксплуатация сети, телефонная служба, функционирование служб и человеческие факторы
Серия F	Нетелефонные службы электросвязи
Серия G	Системы и среда передачи, цифровые системы и сети
Серия H	Аудиовизуальные и мультимедийные системы
Серия I	Цифровая сеть с интеграцией служб
Серия J	**Кабельные сети и передача сигналов телевизионных и звуковых программ и других мультимедийных сигналов**
Серия K	Защита от помех
Серия L	Окружающая среда и ИКТ, изменение климата, электронные отходы, энергоэффективность; конструкция, прокладка и защита кабелей и других элементов линейно-кабельных сооружений
Серия M	Управление электросвязью, включая СУЭ и техническое обслуживание сетей
Серия N	Техническое обслуживание: международные каналы передачи звуковых и телевизионных программ
Серия O	Требования к измерительной аппаратуре
Серия P	Качество телефонной передачи, телефонные установки, сети местных линий
Серия Q	Коммутация и сигнализация, а также соответствующие измерения и испытания
Серия R	Телеграфная передача
Серия S	Оконечное оборудование для телеграфных служб
Серия T	Оконечное оборудование для телематических служб
Серия U	Телеграфная коммутация
Серия V	Передача данных по телефонной сети
Серия X	Сети передачи данных, взаимосвязь открытых систем и безопасность
Серия Y	Глобальная информационная инфраструктура, аспекты межсетевого протокола, сети последующих поколений, интернет вещей и "умные" города
Серия Z	Языки и общие аспекты программного обеспечения для систем электросвязи

Отпечатано в Швейцарии
Женева, 2020 г.