Международный союз электросвязи

МСЭ-Т

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

X.1365
(03/2020)

СЕРИЯ X: СЕТИ ПЕРЕДАЧИ ДАННЫХ,
ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ
И БЕЗОПАСНОСТЬ

Безопасные приложения и услуги (2) –
Безопасность интернета вещей (IoT)

Методика обеспечения безопасности при использовании криптографии на основе идентичности в поддержку услуг интернета вещей (IoT) в сетях электросвязи

Рекомендация МСЭ-Т X.1365

Международный союз электросвязи

РЕКОМЕНДАЦИИ МСЭ-Т СЕРИИ X

СЕТИ ПЕРЕДАЧИ ДАННЫХ, ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ И БЕЗОПАСНОСТЬ

СЕТИ ПЕРЕДАЧИ ДАННЫХ ОБЩЕГО ПОЛЬЗОВАНИЯ	X.1–X.199
ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ	X.200–X.299
ВЗАИМОДЕЙСТВИЕ МЕЖДУ СЕТЯМИ	X.300–X.399
СИСТЕМЫ ОБРАБОТКИ СООБЩЕНИЙ	X.400–X.499
СПРАВОЧНИК	X.500–X.599
ОРГАНИЗАЦИЯ СЕТИ ВОС И СИСТЕМНЫЕ АСПЕКТЫ	X.600–X.699
УПРАВЛЕНИЕ ВОС	X.700–X.799
БЕЗОПАСНОСТЬ	X.800–X.849
ПРИЛОЖЕНИЯ ВОС	X.850–X.899
ОТКРЫТАЯ РАСПРЕДЕЛЕННАЯ ОБРАБОТКА	X.900–X.999
БЕЗОПАСНОСТЬ ИНФОРМАЦИИ СЕТЕЙ	
Общие аспекты безопасности	X.1000–X.1029
Безопасность сетей	X.1030–X.1049
Управление безопасностью	X.1050–X.1069
Телебиометрия	X.1080–X.1099
БЕЗОПАСНЫЕ ПРИЛОЖЕНИЯ И УСЛУГИ (1)	
Безопасность многоадресной передачи	X.1100–X.1109
Безопасность домашних сетей	X.1110–X.1119
Безопасность подвижной связи	X.1120–X.1139
Безопасность веб-среды	X.1140–X.1149
Протоколы безопасности (1)	X.1150–X.1159
Безопасность одноранговых сетей	X.1160–X.1169
Безопасность сетевой идентификации	X.1170–X.1179
Безопасность IPTV	X.1180–X.1199
БЕЗОПАСНОСТЬ КИБЕРПРОСТРАНСТВА	
Кибербезопасность	X.1200–X.1229
Противодействие спаму	X.1230–X.1249
Управление определением идентичности	X.1250–X.1279
БЕЗОПАСНЫЕ ПРИЛОЖЕНИЯ И УСЛУГИ (2)	
Связь в чрезвычайных ситуациях	X.1300–X.1309
Безопасность повсеместных сенсорных сетей	X.1310–X.1319
Безопасность "умных" электросетей	X.1330–X.1339
Сертифицированная электронная почта	X.1340–X.1349
Безопасность интернета вещей (IoT)	**X.1360–X.1369**
Безопасность интеллектуальных транспортных сетей (ИТС)	X.1370–X.1389
Безопасность технологии распределенного реестра	X.1400–X.1429
Безопасность технологии распределенного реестра	X.1430–X.1449
Протоколы безопасности (2)	X.1450–X.1459
ОБМЕН ИНФОРМАЦИЕЙ, КАСАЮЩЕЙСЯ КИБЕРБЕЗОПАСНОСТИ	
Обзор кибербезопасности	X.1500–X.1519
Обмен информацией об уязвимости/состояния	X.1520–X.1539
Обмен информацией о событии/инциденте/эвристических правилах	X.1540–X.1549
Обмен информацией о политике	X.1550–X.1559
Эвристические правила и запрос информации	X.1560–X.1569
Идентификация и обнаружение	X.1570–X.1579
Гарантированный обмен	X.1580–X.1589
БЕЗОПАСНОСТЬ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ	
Обзор безопасности облачных вычислений	X.1600–X.1601
Проектирование безопасности облачных вычислений	X.1602–X.1639
Передовой опыт и руководящие указания в области облачных вычислений	X.1640–X.1659
Обеспечение безопасности облачных вычислений	X.1660–X.1679
Другие вопросы безопасности облачных вычислений	X.1680–X.1699
КВАНТОВАЯ СВЯЗЬ	X.1700–X.1729

Для получения более подробной информации просьба обращаться к перечню Рекомендаций МСЭ-Т.

Рекомендация МСЭ-Т X.1365

Методика обеспечения безопасности при использовании криптографии на основе идентичности в поддержку услуг интернета вещей (IoT) в сетях электросвязи

Резюме

В Рекомендации МСЭ-Т X.1365 содержится методика обеспечения безопасности при использовании криптографии на основе идентичности (IBC) с открытым ключом в поддержку услуг интернета вещей (IoT) в сетях электросвязи, включая механизмы управления определением идентичности, архитектуру управления ключами, операции управления ключами и аутентификацию.

Традиционная методика обеспечения безопасности на основе сертификатов связана с выполнением громоздких операций управления ключами, включая выдачу, запрос и аннулирование (отзыв) сертификатов. Такие системы сталкиваются со значительными трудностями, когда нужно сохранить хорошие рабочие характеристики в условиях растущего числа устройств, подключенных к IoT.

Технология IBC – еще одна методика обеспечения безопасности, в которой в качестве открытого ключа используется идентичность объекта. Важной особенностью IoT является то, что каждая вещь имеет свой уникальный идентификатор (ID). При использовании таких идентификаторов в качестве открытых ключей никакие сертификаты не требуются. Следовательно, система безопасности IBC использует упрощенные операции управления ключами, обеспечивает распределение полномочий для управления собственными устройствами, и она хорошо масштабируется до огромных количеств как конечных точек, так и разнообразных устройств.

Хронологическая справка

Издание	Рекомендация	Утверждение	Исследовательская комиссия	Уникальный идентификатор*
1.0	МСЭ-Т X.1365	26.03.2020 года	17-я	11.1002/1000/14089

Ключевые слова

Безопасность данных пользователей, криптография на основе идентичности, методика обеспечения безопасности, IoT, IBC.

* Для доступа к Рекомендации наберите в адресном поле вашего веб-навигатора URL http://handle.itu.int/, после которого укажите 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 не 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

Рекомендация МСЭ-Т X.1365

Методика обеспечения безопасности при использовании криптографии на основе идентичности в поддержку услуг интернета вещей (IoT) в сетях электросвязи

1 Сфера применения

В настоящей Рекомендации излагается методика обеспечения безопасности при использовании технологии криптографии на основе идентичности (IBC) в поддержку услуг интернета вещей (IoT) в сетях электросвязи. Эта методика охватывает механизмы идентификации устройств, выдачи закрытых ключей и поиска открытых параметров, а также протоколы аутентификации.

ПРИМЕЧАНИЕ. – Данная методика не ограничивается службой IoT, а также может применяться другими службами.

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 настоящей Рекомендации, не придает ему как отдельному документу статус Рекомендации.

[IETF RFC 4764] IETF RFC 4764 (2007), *The EAP-PSK protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method*

[IETF RFC 5091] IETF RFC 5091 (2007), *Identity-Based Cryptography Standard (IBCS) #1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems*

[IETF RFC 5216] IETF RFC 5216 (2008), *The EAP-TLS Authentication Protocol*

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

[IETF RFC 5408] IETF RFC 5408 (2009), *Identity-Based Encryption Architecture and Supporting Data Structures*

[IETF RFC 5480] IETF RFC 5480 (2009), *Elliptic Curve Cryptography Subject Public Key Information*

[IETF RFC 5958] IETF RFC 5958 (2010), *Asymmetric Key Packages*

[IETF RFC 6507] IETF RFC 6507 (2012), *Elliptic Curve-Based Certificateless Signatures for Identity-Based Encryption (ECCSI)*

[IETF RFC 6508] IETF RFC 6508 (2012), *Sakai–Kasahara Key Encryption (SAKKE)*

[IETF RFC 6960] IETF RFC 6960 (2013), *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP*

[IETF RFC 7250] IETF RFC 7250 (2014), *Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)*

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

[ISO/IEC 11770-3] ISO/IEC 11770-3:2015, *Information technology – Security techniques – Key management – Part 3: Mechanisms using asymmetric techniques*

[ISO/IEC 14888-3] ISO/IEC 14888-3:2018, *IT Security techniques – Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms*

[ISO/IEC 18033-5] ISO/IEC 18033-5:2015, *Information technology – Security techniques – Encryption algorithms – Part 5: Identity-based ciphers*

3 Определения

3.1 Термины, определенные в других документах

В настоящей Рекомендации используются следующие термины, определенные в других документах.

3.1.1 поставщик данных идентичности (identity provider) [b-ITU-T Y.2720] – объект, который создает и поддерживает надежную информацию, подтверждающую идентичность других объектов (например, пользователей/абонентов, организаций и устройств), и управляет такой информацией, а также предоставляет базирующиеся на идентичности услуги на основе доверия, деловых отношений и других типов отношений.

3.1.2 идентификатор (identifier (ID)) [b-ITU-T E.101] – последовательность цифр, знаков и символов, используемая для однозначной идентификации абонента, пользователя, элемента сети, функции, объекта сети, услуги или приложения. Идентификаторы могут использоваться для регистрации или санкционирования. Они могут быть либо общего пользования для всех сетей, либо частными для конкретной сети (частные идентификаторы обычно не раскрываются третьим сторонам).

3.1.3 главный открытый ключ (master public key (MPK)) [ISO/IEC 18033-5] – открытое значение, однозначно определяемое соответствующим главным секретным ключом.

3.1.4 главный секретный ключ (master secret key (MSK)) [ISO/IEC 18033-5] – секретное значение, используемое генератором закрытых ключей для вычисления закрытых ключей в алгоритме IBE.

3.1.5 генератор закрытых ключей (private key generator (PKG)) [ISO/IEC 18033-5] – объект или функция, генерирующие набор закрытых ключей.

3.2 Термины, определенные в настоящей Рекомендации

В настоящей Рекомендации определяются следующие термины.

3.2.1 область идентичности (identity domain) – множество объектов, использующих один и тот же набор открытых параметров и правил именования идентичностей.

3.2.2 открытый параметр (public parameter) – один из параметров криптографических вычислений, включая выбор конкретной криптографической схемы или функции из семейства криптографических схем или функций или семейства математических пространств и главного открытого ключа.

3.2.3 сервер открытых параметров (public parameter server) – объект, предоставляющий открытые параметры по запросу.

3.2.4 модуль безопасности (security module (SecM)) – часть программного или аппаратного обеспечения или программно-аппаратного комплекса, надежно реализующая криптографические механизмы и предоставляющая услуги обеспечения безопасности.

4 Сокращения и акронимы

В настоящей Рекомендации используются следующие сокращения и акронимы.

4G	Fourth Generation	Четвертое поколение
5G	Fifth Generation	Пятое поколение
AuC	Authentication Centre	Центр аутентификации
AGW	Aggregate Gateway	Объединительный шлюз
AK	Authentication Key	Ключ аутентификации
AKA	Authenticated Key Agreement	Соглашение об аутентифицированных ключах

AN	Access Node	Узел доступа
AS	Access System	Система доступа
ASN.1	Abstract Syntax Notation one	Абстрактная синтаксическая нотация версии 1
AU	Authentication Unit	Блок аутентификации
BN	Barreto-Naehrig	(кривые) Баррето-Наерига
BLS-12	Barreto-Lynn-Scott embedding degree 12	12-я степень вложения Баррето-Линна-Скотта
BLS-24	Barreto-Lynn-Scott embedding degree 24	24-я степень вложения Баррето-Линна-Скотта
CRL	Certificate Revocation List	Список аннулированных сертификатов
DER	Distinguished Encoding Rules	Отличительные правила кодирования
EAP	Extensible Authentication Protocol	Расширяемый протокол аутентификации
ECCSI	Elliptic Curve-based Certificateless Signatures for Identity-based encryption	Подписи без сертификатов на основе эллиптических кривых для основанного на идентичности шифрования
EID	eUICC-ID	eUICC-ID
EIS	eUICC Information Set	Информационный набор eUICC
eUICC	Embedded Universal Integrated Circuit Card	Встроенная универсальная карта с интегральной схемой
EUM	eUICC Manufacturer	Производитель eUICC
GW	Gateway	Шлюз
HSM	Hardware Security Module	Аппаратный модуль безопасности
HTTP	Hypertext Transfer Protocol	Протокол передачи гипертекста
IBAKA	Identity-Based Authenticated Key Agreement	Соглашение об аутентифицированных ключах на основе идентичности
IBC	Identity-Based Cryptography	Криптография на основе идентичности
IBE	Identity-Based Encryption	Шифрование на основе идентичности
IBS	Identity-Based Signature	Подпись на основе идентичности
ID	Identifier	Идентификатор
IdP	Identity Provider	Поставщик данных идентичности
IMSI	International Mobile Subscription Identity	Международный идентификатор абонента подвижной связи
IoT	Internet of Things	Интернет вещей
ISP	IoT Service Platform	Платформа услуг IoT
IRL	Identity Revocation List	Список аннулированных идентичностей
ISD	Issuer Security Domain	Область безопасности выдавшего органа
KDF	Key Derivation Function	Функция выработки ключей
KDK	Key Derivation Key	Ключ выработки ключей
KEK	Key Encryption Key	Ключ шифрования ключей
KEM	Key Encapsulation Mechanism	Механизм инкапсуляции ключей
KMIP	Key Management Interoperability Protocol	Протокол взаимодействия для управления ключами
KMS	Key Management Service	Служба управления ключами
KPAK	KMS Public Authentication Key	Открытый ключ аутентификации KMS

KSS-16	Kachisa-Schaefer-Scott embedding degree 16	16-я степень вложения Качиса-Шефера-Скотта
KSS-18	Kachisa-Schaefer-Scott embedding degree 18	18-я степень вложения Качиса-Шефера-Скотта
LTE	Long-Term Evolution	Долгосрочное развитие
LTE-M	Long-Term Evolution, category M1	Долгосрочное развитие, категория M1
MAC	Media Access Control	Управление доступом к среде передачи
MNO	Mobile Network Operator	Оператор сети подвижной связи
MSK	Master Secret Key	Главный секретный ключ
NB-IoT	Narrowband Internet of Things	Узкополосный интернет вещей
OCSP	Online Certificate Status Protocol	Онлайновый протокол статуса сертификата
OID	Object Identifier	Идентификатор объекта
OISP	Online Identity Status Protocol	Онлайновый протокол статуса идентичности
PKG	Private Key Generator	Генератор закрытых ключей
PKI	Public Key Infrastructure	Инфраструктура открытых ключей
PPS	Public Parameter Server	Сервер открытых параметров
PVT	Public Verification Token	Открытый маркер проверки
RSF	Revocation Server Function	Функция сервера аннулирования (отзыва)
SecM	Security Module	Модуль безопасности
SK	Sakai-Kasahara	Сакаи-Касахара
SM-DP	Subscription Manager Data Preparation	Функция подготовки данных диспетчера подписки
SM-SR	Subscription Manager Secure Routing	Функция защищенной маршрутизации диспетчера подписки
SOK	Sakai-Ohgishi-Kasahara	Сакаи-Огиси-Касахара
SSK	Secret Signing Key	Секретный ключ подписи
TLS	Transport Layer Security	Безопасность транспортного уровня
TLV	Tag, Length and Vector	Тег, длина и вектор
TVP	Time-Variant Parameter	Зависящий от времени параметр
UE	User Equipment	Оборудование пользователя
UICC	Universal Integrated Circuit Card	Универсальная карта с интегральной схемой

5 Соглашения

Отсутствуют.

6 Обзор

Согласно пункту 6.1 [b-ITU-T Y.4000] IoT "можно рассматривать как глобальную инфраструктуру для информационного общества, которая обеспечивает возможность предоставления более сложных услуг путем соединения друг с другом вещей (физических и виртуальных) на основе существующих и развивающихся функционально совместимых информационно-коммуникационных технологий (ИКТ)". Из-за повсеместного характера распространения устройств в сочетании с возрастающей уязвимостью пользовательских данных одной из главных проблем является обеспечение безопасности IoT. В [b-ITU-T Y.4100] содержится описание общих требований безопасности высокого уровня для IoT, включая безопасность связи, безопасность управления данными, безопасность предоставления услуг, а также взаимную аутентификацию и авторизацию. Кроме того, в [b-ITU-T X.1361]

анализируются угрозы и проблемы безопасности в среде IoT и описываются возможности для их устранения и смягчения. В число требуемых возможностей безопасности, определенных в [b-ITU-T X.1361], входят:

– возможность безопасной связи для поддержки доверенной связи с обеспечением безопасности и защиты конфиденциальности данных;

– возможность безопасного управления ключами для поддержки безопасной связи;

– возможность безопасного управления данными для доверительного управления данными с обеспечением безопасности и защиты их конфиденциальности;

– возможность аутентификации для проверки подлинности устройств;

– возможность авторизации (контроля доступа) для авторизации устройств;

– возможность реализации безопасных протоколов на основе облегченных алгоритмов шифрования.

Устройства IoT характеризуются ограниченностью ресурсов, таких как вычислительные возможности и возможности связи. Природа устройств IoT создает новые препятствия на пути к удовлетворению требований безопасности в системе IoT. В частности, ключевыми факторами при рассмотрении решений безопасности для IoT становятся очень простое развертывание, облегченные операции управления и распределенные полномочия.

Как описано в [b-ITU-T X.1361], к важнейшим услугам, необходимым для защиты IoT, относятся аутентификация, управление доступом и обеспечение целостности и конфиденциальности данных. Для предоставления таких услуг могут использоваться криптографические механизмы как с симметричным ключом, так и с открытым ключом.

Система безопасности на основе симметричного ключа относительно проста. Однако она не подходит для сценариев связи между равноправными объектами, например межмашинных приложений в IoT, без онлайновой службы, играющей роль доверенного посредника, или без предварительного попарного обмена секретными ключами между устройствами. Межсистемная безопасная связь без раскрытия равноправным сторонам секретных ключей пользователя также усложнена.

Традиционное решение шифрования с открытым ключом на основе сертификатов предусматривает тяжелые операции управления ключами, включая выдачу, запрос, распространение, проверку и аннулирование сертификатов. Такие системы сталкиваются со значительными трудностями, когда нужно сохранить хорошие рабочие характеристики в условиях растущего числа устройств и расширения функциональных возможностей IoT. Затраты на обмен сертификатами в протоколах безопасности также вызывают проблемы, особенно в узкополосных сетях интернета вещей (NB-IoT) с малым размером блоков пакетных данных.

Криптография на основе идентичности (IBC) – это еще один тип технологий, использующих в качестве открытого ключа идентичность объекта. Важной особенностью IoT является то, что каждая вещь имеет свой уникальный идентификатор (ID). При использовании таких идентификаторов в качестве открытых ключей никакие сертификаты не требуются. Следовательно, в системе безопасности IBC используются упрощенное управление ключами, распределенные полномочия для управления собственными устройствами, и она хорошо масштабируется до огромных количеств как конечных точек, так и разнообразных устройств. Поскольку сертификаты не передаются, протоколы безопасности могут выполняться эффективнее.

В системе IBC доверенная сторона, называемая службой управления ключами (KMS), отвечает за генерирование закрытого ключа каждого объекта. Прежде чем предоставить услугу генерирования ключа, KMS запускает процесс инициализации системы, вызывая функцию **IBSetup**, которая с учетом параметра безопасности определяет набор системных параметров и генерирует главный секретный ключ (MSK) и главный открытый ключ (MPK). Следует отметить, что KMS имеет ту же функцию, что и генератор закрытых ключей (PKG). Поэтому для удобства в настоящей Рекомендации KMS и PKG используются взаимозаменяемо, и комбинация системных параметров и MPK называется открытыми параметрами. KMS хранит MSK строго конфиденциально и делает открытые параметры общедоступными. При необходимости открытые параметры могут публиковаться специальной службой сервера открытых параметров (PPS).

Типичная система безопасности IBC может использовать для предоставления различных услуг безопасности, включая обеспечение конфиденциальности данных, аутентификацию объектов и установление защищенного канала связи, целый ряд механизмов IBC, таких как шифрование на основе идентичности (IBE), подпись на основе идентичности (IBS) и соглашение об аутентифицированных ключах на основе идентичности (IBAKA). Все эти алгоритмы IBC можно рассматривать как сочетание двух наборов функций. Один набор состоит из функций генерирования ключей, которые создают пары открытого и закрытого ключей на основе идентичности. Функция генерирования закрытого ключа (**IBExtract**) создает закрытый ключ из идентификатора, MSK и открытых параметров. Функция выработки открытого ключа на основе идентичности (**IBDerivate**) вычисляет открытый ключ по идентификатору и открытым параметрам. Другой набор функций, например шифрования или дешифрования (**IBEnc/IBDec**), подписи или проверки подписи (**IBSign/IBVerify**) и протокола установления аутентифицированного сеансового ключа, использует генерированные пары ключей для выполнения соответствующих криптографических операций.

Технология IBC стандартизирована различными организациями по разработке стандартов, в том числе Международной организацией по стандартизации (ИСО), Международной электротехнической комиссией (МЭК), Целевой группой по инженерным проблемам интернета (IETF), Институтом инженеров по электротехнике и радиоэлектронике (IEEE), Европейским институтом стандартизации электросвязи (ЕТСИ) и Управлением по стандартизации Китая (SAC). Список некоторых соответствующих стандартов, разработанных этими организациями, приведен в разделе "Библиография". В выпуске 4 спецификации OneM2M также рассматривается возможность использования технологий IBC для сетей IoT; соответствующий анализ безопасности приведен в [b-ETSI TR 118 508].

В настоящей Рекомендации описывается структура безопасности с использованием технологии IBC для обеспечения возможностей безопасности для услуг IoT в сетях электросвязи. Эта структура охватывает аспекты управления определением идентичности, архитектуру управления ключами, операции управления ключами и аутентификации, а также протоколы соглашения о ключах при использовании IBC.

7 Эталонная архитектура системы для услуг IoT в сетях электросвязи

В данном разделе представлена общая эталонная архитектура системы для услуг IoT, предоставляемых по сетям электросвязи. На рисунке 1 показана концептуальная схема эталонной архитектуры системы для услуг IoT. Эта система состоит из трех частей: устройства IoT, системы доступа (AS) и платформы услуг IoT (ISP).

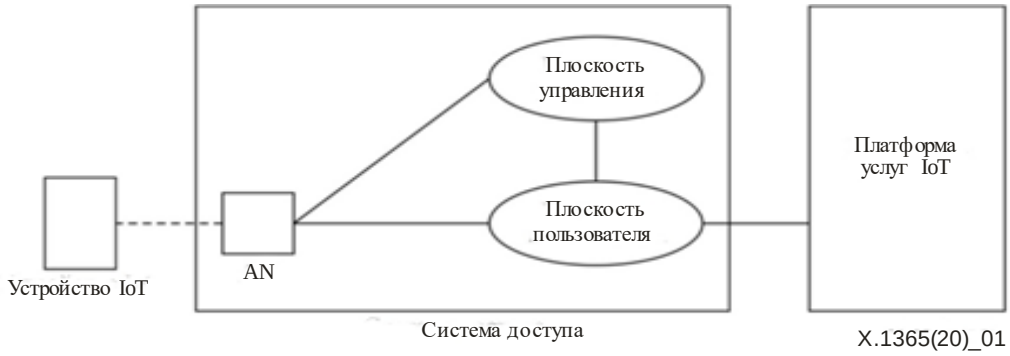

Рисунок 1 – Концептуальная схема архитектуры системы для услуг интернета вещей

Устройства IoT отвечают за сбор данных или выполнение действий. Большинство таких устройств могут устанавливать соединение с системой электросвязи и обмениваться данными с ISP. В настоящее время большинство устройств IoT подключаются к ISP по беспроводному каналу связи, установленному с сетью электросвязи. В настоящей Рекомендации под AS понимается сеть электросвязи. Обычно она состоит из двух частей – сети доступа (AN) и базовой сети. Базовую сеть можно дополнительно подразделить на две части – плоскость управления и плоскость пользователя, отвечающие соответственно за сигнализацию управления и передачу данных.

Сеть электросвязи, как традиционное беспроводное соединение, уже используется на протяжении нескольких поколений. Исторически сети электросвязи предназначались для поддержки подвижной связи между людьми с возможностью беспрепятственного роуминга. В последние годы, начиная с сетей четвертого поколения 4G-LTE (долгосрочное развитие), в их конструкции также учитывается поддержка устройств IoT. Например, в сети 4G-LTE для поддержки устройств IoT разработаны технологии категории M1 (LTE-M) и NB-IoT.

Большинство современных систем электросвязи состоит из трех компонентов: терминалов или оборудования пользователя (UE), AN и базовых сетей. Здесь предполагается, что как AN, так и базовые сети принадлежат к AS, как показано на рисунке 1. Услуги IoT обычно находятся за пределами сетей электросвязи, и для передачи данных и управления услугами используются некоторые интерфейсы. Чтобы обеспечить лучшую поддержку услуг IoT, в системную спецификацию сетей электросвязи включают дополнительную специфику IoT. В последние годы интеграция между сетями электросвязи и услугами IoT стала более тесной.

Благодаря системным спецификациям, разработанным для сетей пятого поколения (5G), для услуг IoT поддерживаются технологии открытых ключей, включая аутентификацию для получения доступа к сети. Как указано в разделе 6, по сравнению с другими технологиями открытых ключей IBC проще в управлении и обеспечивает эффективную передачу данных. Поэтому для использования IBC для услуг IoT в сетях электросвязи требуется спецификация в виде стандарта, дополняющего существующие спецификации.

8 Структура применения криптографии на основе идентичности для услуг IoT в сетях электросвязи

В данном разделе представлена структура применения технологий открытых ключей IBC для услуг IoT в сетях электросвязи. Эта структура охватывает системную архитектуру, включая сетевые компоненты, необходимые при использовании технологий IBC. Кроме того, определена структура управления ключами для IBC, поскольку это важная часть системы, использующей технологию IBC. В рамках этой структуры также рассматриваются другие критически важные вопросы, такие как управление ключами, именование идентичностей и протоколы аутентификации.

8.1 Архитектура системы IoT с использованием криптографии на основе идентичности

Для услуг IoT, предоставляемых по сетям электросвязи, IBC может использоваться в целях аутентификации для получения доступа к сети либо к услугам или и того и другого. Аутентификация для получения доступа к сети позволяет определить, можно ли предоставить устройству доступ к сети, а аутентификация для получения доступа к услугам – можно ли предоставить устройству доступ к ISP.

Устройства IoT могут получить прямой или опосредованный доступ к сети электросвязи. Следовательно, существуют две модели доступа:

– модель с прямым соединением – устройства IoT напрямую соединяются с AS;

– модель с опосредованным соединением – устройства IoT соединяются с AS через объединительный шлюз (AGW).

На рисунке 2 показана эталонная архитектура системы IoT, в которой IBC используется для защиты безопасности как AS, так и ISP. С точки зрения безопасности как AS, так и ISP могут предъявлять собственные требования к услугам IoT. Учитывая, что удостоверения безопасности могут предоставлять либо AS, либо ISP, существуют три сценария использования IBC для сетей IoT.

– Сценарий использования IBC для защиты безопасности AS

  В этом сценарии AS выдает удостоверения безопасности для доступа к сети, хранящиеся в устройствах IoT, и управляет ими. При подключении устройств IoT к AS они аутентифицируются ею. Например, устройство IoT вычисляет подпись IBS на основе закрытого ключа, предоставленного AS, и отправляет эту подпись в AS. Соответственно AS может аутентифицировать устройство IoT на основе подписи IBS, передаваемой в сообщениях аутентификации. Если проверка прошла успешно, AS передает данные, полученные от устройства IoT, на сервер IoT.

– Сценарий использования IBC для защиты безопасности ISP

ISP выдает удостоверения безопасности IBC для доступа к услугам, хранящиеся в устройствах IoT, и управляет ими. ISP аутентифицирует устройства IoT на основе подписи, созданной с использованием удостоверений IBC.

– Сценарий использования IBC для защиты безопасности как AS, так и ISP

Удостоверения безопасности IBC, хранящиеся в устройствах IoT, предоставляются и управляются либо AS, либо ISP, либо обоими совместно. Как AS, так и ISP могут аутентифицировать устройство IoT по одним и тем же удостоверениям.

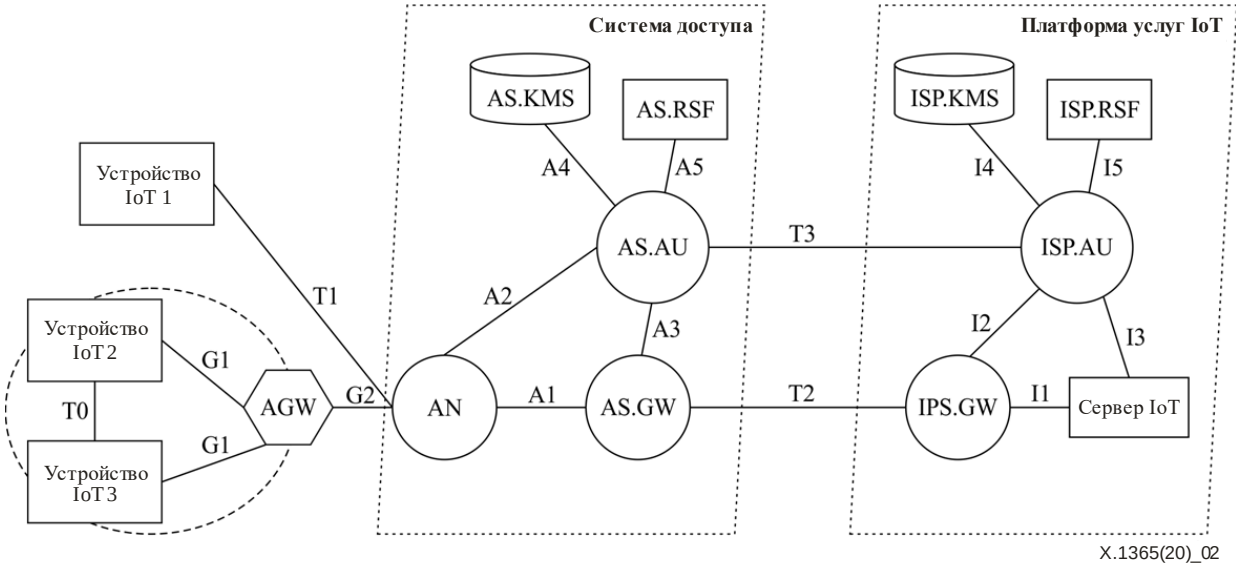

Рисунок 2 – Архитектура системы IoT с использованием криптографии на основе идентичности в сценариях защиты безопасности как системы доступа, так и платформы услуг IoT

Три описанных выше сценария охватывают большинство случаев использования IBC для доступа к сети и услугам. Однако могут быть и другие сценарии, которые выходят за рамки настоящей Рекомендации.

Архитектура системы IoT на основе IBC состоит из следующих сетевых функций (NF) и устройств:

– система доступа (AS) – система доступа устройств IoT или AGW, включающая узел доступа (AS.AN), функцию системы управления ключами (AS.KMS), блок аутентификации (AS.AU), функцию сервера аннулирования (AS.RSF) и шлюз (AS.GW);

– платформа услуг IoT (ISP) – платформа управления услугами IoT, включающая функцию системы управления ключами (ISP.KMS), блок аутентификации (isp.au), функцию сервера аннулирования (ISP.RSF), шлюз (ISP.GW) и сервер IoT. ISP должна поддерживать управление ключами, их распределение, аутентификацию идентификационных данных, шифрование или дешифрование и подпись или проверку подписи и т. д.;

– объединительный шлюз (AGW) – агрегирующий узел, который отвечает за подключение устройств IoT, агрегирование данных всех устройств IoT и их передачу в AS. AGW играет роль посредника при передаче данных между устройствами IoT и AN;

– узел доступа (AN) – узел доступа для устройств IoT или AGW может быть точкой доступа беспроводной или фиксированной сети;

– функция системы управления ключами (KMS) – система управления, отвечающая за генерирование, распределение и обновление ключей IBC и параметров устройств IoT и сетевых функций;

– блок аутентификации (AU) – AU аутентифицирует устройство IoT на основе системы IBC;

– функция сервера аннулирования (RSF) – сервер ведет список аннулированных идентичностей (IRL). Открытые ключи или идентичности, внесенные в список аннулирования, исключаются из использования;

ПРИМЕЧАНИЕ. – У AS и ISP могут быть собственные KMS, AU и RSF.

– шлюз системы доступа (AS.GW) – сетевой элемент, подключенный к GW IoT, который отвечает за передачу пользовательских данных IoT;

– шлюз IoT (IoT GW) – GW, отвечающий за пересылку или агрегирование данных и передачу данных на сервер IoT или за пересылку данных/сигналов от сервера IoT в устройства IoT;

– сервер IoT – сервер, расположенный на стороне поставщика услуг IoT, собирающий данные IoT от GW IoT;

– устройство IoT – конечное устройство, используемое для сбора данных и установления соединения с AN и сервером IoT, которое обеспечивает защиту данных, включая согласование ключей, шифрование или дешифрование и подпись или проверку подписи.

Функции контрольных точек, показанных на рисунке 2, описываются следующим образом:

– G1 – контрольная точка между устройством IoT и AGW, используемая для аутентификации и передачи данных по безопасности;

– G2 – контрольная точка между AGW и AN, используемая для сигнализации и передачи данных между AGW и AN;

– T0 – контрольная точка между устройствами IoT, используемая для сигнализации и обмена данными;

– T1 – контрольная точка между устройствами IoT и AN, используемая для аутентификации и передачи данных по безопасности;

– T2 – контрольные точки между AS.GW и ISP.GW, обеспечивающие туннель данных в плоскости пользователя между AS.GW и ISP;

– T3 – контрольные точки между AS.AU и ISP.AU для обмена сигналами, включая обмен идентификационными данными или выдачу ключей;

– A1 – контрольные точки между AN и AS.GW для туннелирования данных плоскости пользователя;

– A2 – контрольные точки между AS.AU и AN для сигнализации в плоскости управления;

– A3 – контрольные точки между AS.AU и AS.GW для протокола распределения ресурсов и управления GW в AS;

– A4 – контрольные точки между AS.AU и AS.KMS для протокола выдачи ключей в AS;

– A5 – контрольные точки между AS.AU и AS.RSF для протокола аннулирования идентичности или ключей в AS;

– I1 – контрольные точки между сервером IoT и ISP.GW для туннелирования данных плоскости пользователя;

– I2 – контрольные точки между ISP.AU и ISP.GW для протокола распределения ресурсов и управления GW в ISP;

– I3 – контрольные точки между ISP.AU и сервером IoT для обмена такой информацией, как информация о подписке на услуги, передаваемая из сервера IoT в ISP.AU, или сообщения с уведомлением об аутентификации, передаваемые из ISP.AU на сервер IoT;

– I4 – контрольные точки между ISP.AU и ISP.KMS для протокола выдачи ключа в ISP;

– I5 – контрольные точки между ISP.AU и ISP.RSF для протокола аннулирования идентичности или ключей в ISP.

8.2 Архитектура управления ключами

В данном разделе описывается функциональная архитектура, необходимая для поддержки управления ключами при использовании механизмов IBC в IoT. В зависимости от того, имеет ли устройство IoT встроенную универсальную карту с интегральной схемой (eUICC) [b-GSMA SGP.02], рассматриваются

два типа архитектуры управления ключами с использованием IBC: 1) в устройствах IoT с eUICC; и 2) в устройствах IoT без eUICC.

При использовании IBC в устройствах IoT с eUICC архитектура соответствует общей архитектуре удаленного предоставления ресурсов eUICC, определенной в [b-GSMA SGP.02], с добавлением двух новых функциональных объектов, таких как KMS и PPS. В зависимости от местонахождения KMS этот случай дополнительно делится на два следующих:

1) KMS управляется организацией, которая отвечает также за оператора сети подвижной связи (MNO), см. рисунок 3;

2) KMS управляется объектом, отвечающим за подготовку данных диспетчера подписки (SM-DP), см. рисунок 4.

В обоих случаях ключи, в том числе закрытый ключ и открытые параметры, генерируются тогда, когда MNO размещает заявку на профиль. Затем ключи передаются в удаленные устройства eUICC, поскольку эти ключи установлены в соответствии с действующей спецификацией удаленной выдачи ключей [b-GSMA SGP.02]. Подробная информация о ролях, соответствующих функциях и интерфейсах удаленной выдачи ключей eUICC приведена в [b-GSMA SGP.02]. Спецификация профиля, формата хранения и использования этих ключей в eUICC выходит за рамки настоящей Рекомендации.

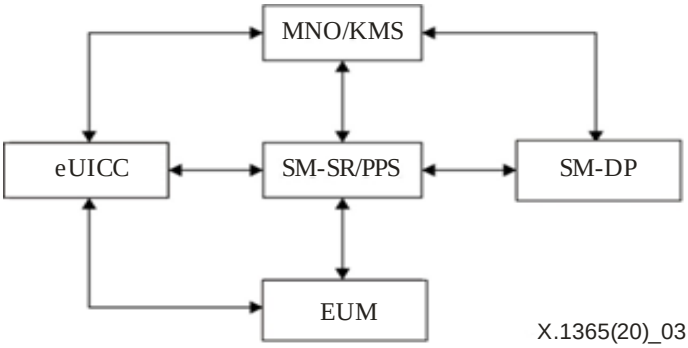

Рисунок 3 – Архитектура A управления криптографическими ключами на основе идентичности для устройств IoT со встроенной универсальной картой с интегральной схемой

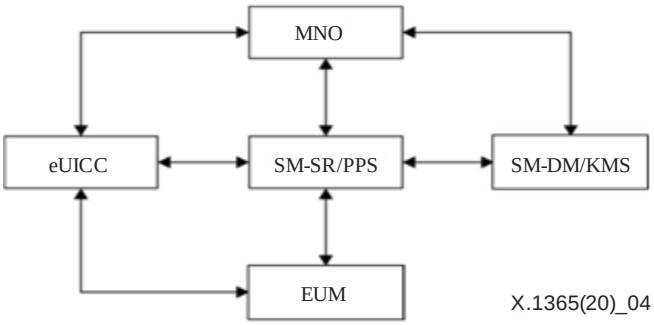

Рисунок 4 – Архитектура B управления криптографическими ключами на основе идентичности для устройств IoT со встроенной универсальной картой с интегральной схемой

Общая архитектура для случаев использования IBC в устройствах IoT без eUICC показана на рисунке 5. Имеются следующие конструктивные блоки:

– SecM – модуль безопасности (SecM), который представляет собой элемент, способный безопасно хранить ключи и реализовывать механизмы безопасности с сохраненными ключами для выполнения операций безопасности. Устройство IoT должно иметь SecM;

– IdP – поставщик данных идентичности (IdP), представляющий собой объект, который создает и поддерживает информацию, подтверждающую идентичность, и управляет такой информацией;

– AuC – центр аутентификации (AuC), который предлагает услугу аутентификации объектов.

Для аутентификации устройств IoT IdP полагается на услугу аутентификации, предоставляемую AuC. После начального процесса аутентификации IdP предоставляет SecM услугу обеспечения идентичности, включая создание, присвоение, замену и аннулирование идентичности. После создания и присвоения устройству IoT новой идентичности IdP вызывает услугу генерирования секретного ключа, предоставляемую KMS, для создания секретного ключа, соответствующего вновь присвоенному идентификатору, и безопасно передает ключи SecM. IdP также извлекает из KMS открытые параметры и передает их на PPS, который публикует эти открытые параметры для внешних объектов. IdP также может предоставлять услуги аутентификации другим объектам, выполняя специальные протоколы аутентификации SecM, включая те, что определены в настоящей Рекомендации.

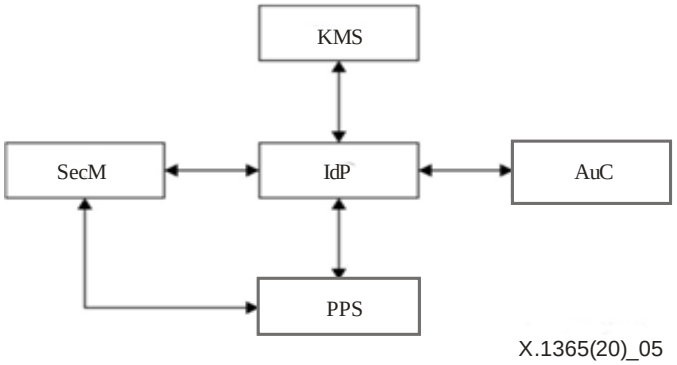

Рисунок 5 – Архитектура управления криптографическими ключами на основе идентичности для устройств IoT без встроенной универсальной карты с интегральной схемой

8.3 Именование идентичностей

При использовании технологии IBC для услуг IoT в сети электросвязи именование идентичности может обеспечить полезную информацию, помогающую операторам управлять сетью. В идентификационные данные (идентичность) могут быть включены различные элементы информации, такие как тип услуг, местоположение, идентификатор устройства и действительное время. Часть этой информации, например действительное время, необходима при использовании технологии IBC. С помощью информации, подтверждающей идентичность, оператор может оптимизировать управление сетью, например, путем выделения соединения определенным сегментам сети в зависимости от типа услуг. Устройство также легко найти по информации о его местоположении. Пример определения идентичности приводится в Дополнении I.

8.4 Управление ключами

Помимо значения идентичности система IBC включает три типа значений криптографических ключей – MSK, открытые параметры и закрытый ключ. Определение этих структур ключей в формате абстрактной синтаксической нотации версии 1 (ASN.1) дано в Приложении B.

Для управления этими ключами в системе IBC используются следующие пять операций управления ключами:

1) инициализация системы;
2) инициализация устройства;
3) поиск открытых параметров;
4) предоставление идентичности и ключей;
5) аннулирование идентичности и ключей.

Для обмена сообщениями между управляющим объектом и KMS может использоваться протокол взаимодействия для управления ключами (KMIP) [b-OASIS KMIP]. Однако требуется определение расширения, необходимого для того, чтобы KMIP удовлетворял новым требованиям функций **IBSetup** и **IBExtract**. Для устройств IoT с eUICC используются стандартные процедуры удаленной выдачи

ключей [b-GSMA SGP.02]. Для устройств IoT без eUICC протоколы взаимодействия между SecM и объектами управления определяются на основе протокола передачи гипертекста (HTTP). Спецификации этих операций определены в Приложении C.

Операция инициализации системы инициализирует систему IBC путем генерирования MSK и открытых параметров. Предполагается, что за процесс инициализации системы IBC отвечает управляющий объект, такой как IdP, SM-DP или MNO. Он устанавливает безопасный канал связи с объектом KMS, реализующим функцию **IBSetup**. Обе стороны выполняют KMIP с помощью операции создания пары ключей. Управляющий объект предоставляет KMS необходимую информацию для вызова функции **IBSetup** и генерирования MSK и открытых параметров. KMIP дополнен поддержкой функций настройки, включая функции различных стандартизированных алгоритмов IBC. Подробная информация об этой операции приведена в разделе C.1 Приложения C.

Операция инициализации устройства заключается в подготовке устройства IoT к предоставлению идентичности и ключей. Возможны два случая – инициализация устройств IoT с eUICC и инициализация устройств IoT без eUICC. Для устройств с eUICC требуется, чтобы eUICC выполнила регистрацию в функции защищенной маршрутизации диспетчера подписки (SM-SR) и была готова к загрузке профиля [b-GSMA SGP.02]. Для стандартных устройств eUICC дополнительные операции не требуются. Для устройств IoT без eUICC SecM сначала следует зарегистрировать в AuC, чтобы получить идентификатор передачи (PROV.ID) и удостоверение передачи (PROV.CRED). Эта пара PROV.ID/PROV.CRED используется для аутентификации объекта в процессе предоставления идентичности/ключей. Если устройства IoT не могут установить безопасный канал связи с IdP с использованием протокола безопасности транспортного уровня (TLS), то требуется, чтобы в процессе инициализации устройства в SecM был дополнительно установлен идентификатор ключа IdP.ID и соответствующий открытый ключ IdP.PUK, принадлежащий IdP, или открытый параметр. Подробная информация об этой операции приведена в разделе C.2.

Операция поиска открытых параметров заключается в извлечении открытых параметров IBC. Устройство IoT использует процедуру предоставления идентичности и ключей для получения открытых параметров системы IBC, к которой оно принадлежит. Для извлечения открытых параметров другой системы IBC из известного PPS оно может следовать спецификации, определенной в разделе 4 [IETF RFC 5408]. Подробная информация об этой операции приведена в разделе C.3.

Операция предоставления идентичности и ключей включает процедуру присвоения идентичности, извлечения закрытого ключа и распределения ключей. После процесса инициализации в устройствах IoT имеются только предварительные идентификационные данные. IdP, SM-DP или MNO должны определить, какую идентичность следует присвоить запрашивающему устройству, а затем связаться с KMS, чтобы сгенерировать соответствующий закрытый ключ и, наконец, безопасно передать устройству идентификационные данные, закрытый ключ и открытые параметры. Подробная информация об этой операции приведена в разделе C.4.

Операция аннулирования идентичности и ключей используется, когда строгая политика безопасности требует своевременного аннулирования идентичности. Если идентичность аннулирована, ей присваивается статус аннулированной. В случае если запись запрашивает статус аннулированной идентичности, IdP, SM-DP или MNO направляют в ответ правильное значение, определенное в онлайновом протоколе статуса идентичности (OISP). Для более эффективной проверки целого пакета статусов идентичности можно регулярно извлекать IRL из IdP, SM-DP или MNO и сохранять его локально, а затем проверять по новому IRL, аннулирована ли идентичность, не запрашивая информацию о статусе в онлайновом режиме для каждой идентичности. Подробная информация об этой операции приведена в разделе C.5.

8.5 Аутентификация

Аутентификация – это процесс определения того, имеет ли объект (устройство или пользователь) право доступа к определенным ресурсам. В сетях электросвязи в отношении устройств IoT применяется аутентификация двух типов – аутентификация для получения доступа к сети и аутентификация для получения доступа к услугам. Аутентификация для получения доступа к сети позволяет определить, можно ли предоставить устройству доступ к сети, а аутентификация для получения доступа к услугам – можно ли предоставить устройству доступ к ISP.

Протоколы аутентификации, построенные на основе технологий IBC, подходят для аутентификации IoT в сетях электросвязи. Это связано с тем, что IBC может значительно снизить нагрузку по управлению определением идентичности и управлению ключами при огромном количестве устройств IoT. Другое преимущество IBC состоит в том, что она обеспечивает распределенную аутентификацию, которая не только значительно сокращает время аутентификации, но и допускает новые сценарии применения, такие как аутентификация между устройствами или аутентификация между транспортными средствами. В современных сетях электросвязи, таких как сети 4G LTE, IBC можно использовать при аутентификации между устройствами IoT и ISP. В сотовых сетях 5G IBC можно использовать как в целях аутентификации для получения доступа к сети, так и в целях аутентификации для получения доступа к услугам. В действующей спецификации безопасности 5G [b-ETSI TS 133.501] определяется унифицированная система аутентификации, поддерживающая методы расширяемого протокола аутентификации (EAP). В приложении к [b-ETSI TS 133.501] дополнительно указано, как использовать EAP-TLS для IoT в сетях 5G.

Система EAP является открытой и поддерживает множество протоколов аутентификации, включая EAP-TLS. Методы аутентификации EAP поддерживают как симметричные, так и асимметричные ключи.

Будучи относительно новой технологией открытых ключей, IBC не имеет надлежащей поддержки в существующих протоколах аутентификации. Поэтому в Приложении D в четыре существующих протокола внесены дополнения для обеспечения поддержки IBC при аутентификации:

1) раздел D.1. Однопроходный протокол передачи секретных ключей [ISO/IEC 11770-3];
2) раздел D.2. TLS с открытым ключом без дополнительной обработки [IETF RFC 8446];
3) раздел D.3. EAP-TLS [IETF RFC 5216];
4) раздел D.4. EAP-PSK [IETF RFC 4764].

9 Требования безопасности

В настоящей Рекомендации рассматриваются только требования безопасности при использовании IBC в IoT. Общие угрозы и требования безопасности для IoT приведены в [b-ITU-T X.1361]. Первостепенными задачами безопасности криптосистемы являются обеспечение целостности и подлинности используемых открытых ключей, а также секретность используемых долгосрочных и временных секретных ключей. В систему IBC входят следующие компоненты: MSK, открытые параметры, идентификаторы, закрытые ключи и временные секретные ключи, используемые в криптографических операциях.

9.1 Требования безопасности в отношении главного секретного ключа

Все закрытые ключи генерируются MSK. В частности, в случае взлома MSK злоумышленник сможет воссоздать закрытый ключ любой идентичности и, следовательно, расшифровать все сообщения, защищенные соответствующим открытым ключом, или выдать себя за любой объект. Любой несанкционированный доступ к MSK может поставить под угрозу безопасность системы IBC. Следовательно, MSK должен храниться в защищенной среде, такой как аппаратный модуль безопасности (HSM). Каждый доступ к ключу должен аутентифицироваться с использованием надежных механизмов безопасности.

9.2 Требования безопасности в отношении открытых параметров

Открытый ключ вычисляется по открытым параметрам и идентификатору с помощью операции **IBDerivate**. Следовательно, использование ложного набора открытых параметров, сгенерированных злоумышленником, для шифрования сообщения или проверки подписи приведет к нарушению секретности зашифрованного сообщения или к подлогу подписи. Поэтому открытые параметры должны передаваться по безопасному каналу или с действительной подписью. Прежде чем принять открытые параметры, объект должен проверить обратившийся к нему объект по безопасному каналу или проверить подлинность подписи по доверенному открытому ключу.

9.3 Требования безопасности к идентификатору

В IoT каждый объект имеет собственный идентификатор. Если один и тот же идентификатор присвоен нескольким объектам и каждому объекту выдан соответствующий закрытый ключ, это может привести к утечке конфиденциальной информации или атакам путем подмены объекта. Поэтому каждому устройству должен быть присвоен уникальный идентификатор.

9.4 Требования безопасности к закрытому ключу

При взломе системы безопасности устройства IoT закрытый ключ может быть раскрыт. Поэтому закрытый ключ должен распределяться по защищенному каналу и храниться в защищенной среде.

9.5 Требования безопасности к временным секретным ключам

При взломе системы безопасности устройства IoT временные секретные ключи, такие как случайный секретный ключ, используемый в процессах шифрования или подписи, могут быть раскрыты. Поэтому временные ключи должны быть гарантированно случайными.

Приложение A

Общая формулировка и алгоритмы криптографии на основе идентичности

(Данное Приложение является неотъемлемой частью настоящей Рекомендации)

В этом Приложении дается общая формулировка IBC и приводится список алгоритмов IBC, которые поддерживаются в настоящей Рекомендации. Алгоритмы, которые соответствуют этой общей формулировке, но не указаны в нижеследующем списке, также могут быть легко включены в эту структуру в будущем в качестве расширений. Приведенная здесь общая формулировка также служит основой для описания соответствующих структур данных ключей, операций управления ключами, а также протоколов аутентификации и создания ключей, определенных в Приложениях B–D.

Криптосистема IBC включает следующие типы данных ключей, причем категоризация этих ключей соответствует [ISO/IEC 18033-5]:

– *ib.msk* – MSK представляет собой секретное значение, используемое KMS для вычисления закрытого ключа на основе идентичности. Значение *ib.msk* генерируется в процессе инициализации системы и известно только KMS;

– *ib.mpk* – MPK, однозначно определяемый соответствующим MSK. *ib.mpk* вычисляется KMS в процессе инициализации системы;

– *ib.sysparam* – системные параметры для криптографических вычислений, включая выбор конкретной криптографической схемы или функции из семейства криптографических схем или функций или семейства математических пространств. *ib.sysparam* выбирается KMS в процессе инициализации системы;

– *ib.pubparam* – открытые параметры, которые представляют собой сочетание системных параметров *ib.sysparam* с MPK *ib.mpk*. Этот тип ключей определен для обеспечения унифицированного представления в международных стандартах, таких как [ISO/IEC 18033-5] и RFC, связанных с IBC, таких как [IETF RFC 5091];

– *ib.prk* – закрытый ключ на основе идентичности, генерируемый KMS с использованием *ib.msk* и *ib.pubparam*, который соответствует идентификатору ID;

– *ib.pub* – открытый ключ на основе идентичности, вычисляемый по идентификатору ID и параметру *ib.pubparam* с помощью функции, определенной криптографической схемой на основе идентичности.

Криптосистема IBC может включать следующие функции, для которых указаны входы и выходы.

IBSetup

вход: параметр безопасности

выход: *ib.pubparam, ib.msk*

IBExtract

вход: *ib.pubparam, ib.msk, ID*

выход: *ib.prk*

IBDerivate

вход: *ib.pubparam, ID*

выход: *ib.puk*

IBEnc

вход: *ib.pubparam, ID,* сообщение *M*

выход: шифрованный текст *C*

IBDec

вход: *ib.pubparam, ID, ib.prk,* шифрованный текст *C*

выход: открытый текст *M* или ошибка

IBSign

вход: *ib.pubparam, ID, ib.prk,* сообщение *M*

выход: подпись *S*

IBVerify

вход: *ib.pubparam, ID,* сообщение *M,* подпись *S*

выход: действительно или ложно

Настоящая Рекомендация поддерживает использование следующих алгоритмов на основе идентичности:

– BB1-KEM (механизм инкапсуляции ключей (KEM)) [IETF RFC 5091];

– BF-IBE [IETF RFC 5091];

– SK-KEM [IETF RFC 6508];

– SM9-IBE [b-GM/T 0044.2];

– Cha-Cheon-IBS (IBS2) [ISO/IEC 14888-3];

– ECCSI (подписи без сертификатов на основе эллиптических кривых для основанного на идентичности шифрования) [IETF RFC 6507];

– Hess-IBS (IBS1) [ISO/IEC 14888-3];

– SM9-IBS (китайская IBS) [ISO/IEC 14888-3];

– Fujioka-Suzuki-Ustaoglu-AKA (соглашение об аутентифицированном ключе (AKA)) [ISO/IEC 11770-3];

– Smart-Chen-Cheng-AKA [ISO/IEC 11770-3];

– SM9-AKA [b-GM/T 0044.2];

– Wang-AKA [b-IEEE P1363.3].

Все эти алгоритмы основаны на предположении о дискретном логарифме и, как правило, реализуются в группе точек на эллиптической кривой. Многие из этих алгоритмов также используют криптографическое спаривание (образование криптографических пар) на эллиптической кривой [b-Galbraith]. Криптографическое спаривание *e* представляет собой эффективно вычисляемое билинейное преобразование *e*: $G1 \times G2 \rightarrow G3$, удовлетворяющее уравнению

$$e([a]P1, [b]P2) = e(P1, P2)^{a*b},$$

где P1 и P2 – генератор циклической группы соответственно *G*1 и *G*2. [*a*]P1 означает *a* групповых операций с P1, аналогично [*b*]P2 – групповая операция с P2.

Криптографическое спаривание может быть реализовано путем образования пар по Вейлю или Тейту, оптимального спаривания по Атэ и т. д. на подходящих для этого эллиптических кривых [b-Freeman]. Обычно к подходящим для образования пар эллиптическим кривым относят суперсингулярные эллиптические кривые, кривые Баррето-Наерига (BN), кривые 12-й степени вложения Баррето-Линна-Скотта (BLS-12), кривые 16-й степени вложения Качиса-Шефера-Скотта (KSS-16), кривые 18-й степени вложения Качиса-Шефера-Скотта (KSS-18) и кривые 24-й степени вложения Баррето–Линна–Скотта (BLS-24) [b-Freeman]. Все эти кривые *E* основаны на простом поле, конечном поле простой характеристики p, F_p, где p – простое целое число. *G*1 – это подгруппа точек на кривой *E*. *G*2 либо совпадает с *G*1, если используются суперсингулярные кривые, либо представляет собой подгруппу точек на кривой кручения *E*. *E* строится из некоторого поля расширения базового поля F_p. *G*3 – это расширение поля F_p^k базового поля F_p, где k – степень вложения.

Алгоритмы IBC построены на других математических механизмах, таких как решетки, см., например, [b-Ducas]. Алгоритм этого типа эффективен в отношении вычислений, но имеет больший размер ключа и выходных данных по сравнению с алгоритмами, основанными на дискретном логарифме по эллиптическим кривым. Считается, что эти алгоритмы будут устойчивы к атакам, реализуемым на квантовых компьютерах. Однако алгоритмы этой категории еще находятся в стадии разработки. Следовательно, рассмотрение возможности их стандартизации представляется преждевременным, но алгоритмы IBC, основанные на решетке, могут быть рассмотрены на предмет их включения в будущем.

Приложение B

Спецификация данных криптографических ключей на основе идентичности

(Данное Приложение является неотъемлемой частью настоящей Рекомендации)

В [IETF RFC 5408] определена общая структура системных параметров с использованием стандартного метода ASN.1, включая *ib.pubparam* и другую вспомогательную информацию, а в [IETF RFC 5091] определены два набора структур данных ключей, в том числе *ib.msk* и *ib.prk,* для двух алгоритмов IBE, то есть BF-IBE и BB1-IBE. Сохраняя совместимость с существующими определениями, настоящая Рекомендация расширяет определение системных параметров и определяет новые структуры данных ключей для поддержки дополнительных алгоритмов и различных эффективных реализаций с различными кривыми и видами спаривания.

Общая структура системных параметров определяется следующим образом.

```
IBSysParams ::= SEQUENCE {
        version                 INTEGER { v3(3) },
        domainName              IA5String,
        domainSerial            INTEGER,
        validity                ValidityPeriod,
        ibPublicParameters      IBPublicParameters,
        ibIdentityType          OBJECT IDENTIFIER,
        ibParamExtensions       [0] IMPLICIT IBParamExtensions OPTIONAL,
        signatureAlgorithm      [1] IMPLICIT AlgorithmIdentifier OPTIONAL,
        signature               [2] IMPLICIT BIT STRING OPTIONAL
}
```

IBSysParams соответствует определению IBESysParams, приведенному в [IETF RFC 5408], но версия (version) изменена на v3(3), и добавлены два дополнительных поля. Параметры districtName и districtSerial переименованы соответственно в domainName и domainSerial. В определении IBPublicParameter тип OCTET STRING заменен на вновь определенный тип IBParameterData, который представляет собой CHOICE, определяемый значением pkgAlgorithm. Это определение устраняет излишнее двойное кодирование, вызванное предыдущим определением, а именно кодирование publicParameterData как SEQUENCE, например параметров BFPublicParameter, и дальнейшее кодирование результата как OCTET STRING. За исключением двух новых полей, значения других полей остаются неизменными, такими как в [IETF RFC 5408]. Два новых поля имеют следующие значения:

– signature algorithm – это алгоритм подписи, используемый для генерирования значения подписи. Это необязательное поле, поскольку поле подписи не является обязательным;

– поле signature содержит цифровую подпись, вычисленную по результату применения отличительных правил кодирования (DER) ASN.1 из версии поля в *ibParamExtensions*. Это поле кодируется как BIT STRING и не является обязательным.

Поле подписи, если оно используется, помогает объекту проверить подлинность открытых параметров системы, не прибегая к другим методам. Например, если устройство IoT не имеет возможности установить защищенный канал на основе TLS для извлечения открытых параметров другой системы IBC, как того требует [IETF RFC 5408], оно может запросить ее PPS через HTTP. В этом случае обслуживающий PPS подписывает запрошенные открытые параметры своим закрытым ключом подписи. Устройство IoT может проверить подпись, чтобы убедиться в подлинности ответа. Если PPS публикует открытые параметры другой системы IBC для обслуживаемых им объектов, рекомендуется, чтобы сообщение с подписью обрабатывалось как идентичность, а алгоритм **IBExtract** использовался как алгоритм подписи для генерирования закрытого ключа в качестве соответствующего значения

подписи. Таким образом устройства IoT проверяют, является ли значение подписи действительным закрытым ключом, соответствующим результату преобразования DER ASN.1 из версии поля в *ibParamExtensions*, и не нуждаются в дополнительном проверочном открытом ключе для проверки подписи:

```
ValidityPeriod ::= SEQUENCE {
        notBefore       GeneralizedTime,
        notAfter        GeneralizedTime
}
IBPublicParameters ::= SEQUENCE SIZE (1..MAX) OF IBPublicParameter
IBPublicParameter ::= SEQUENCE {
        pkgAlgorithm            OBJECT IDENTIFIER,
        publicParameterData     IBParameterData
}
```

Значение publicParameterData определяется параметром pkgAlgorithm. Оно может быть одним из следующих возможных вариантов:

```
IBParameterData ::= CHOICE{
        bb1ParameterData        [0] IMPLICIT BB1PublicParameters,
        bfParameterData         [1] IMPLICIT BFPublicParameters,
        eccsiParameterData      [2] IMPLICIT ECCSIPublicParameters,
        skParameterData         [3] IMPLICIT SKPublicParameters,
        sm9ParameterData        [4] IMPLICIT SM9PublicParameters
}
IBParamExtensions ::= SEQUENCE OF IBParamExtension
IBParamExtension ::= SEQUENCE {
        ibParamExtensionOID             OBJECT IDENTIFIER,
        ibParamExtensionValue           OCTET STRING
}
AlgorithmIdentifier  ::= SEQUENCE {
        algorithm               OBJECT IDENTIFIER,
        parameters              ANY DEFINED BY algorithm OPTIONAL
}
```

В [IETF RFC 5091] определены два набора MSK, открытые параметры и блок закрытых ключей, то есть:

– BB1MasterSecret, BB1PublicParameters, BB1PrivateKeyBlock;

– BFMasterSecret, BFPublicParameters, BFPrivateKeyBlock определены для функции генерирования ключей BF и BB1. Эти назначения подходят только для реализаций функций с симметрическим спариванием на суперсингулярных эллиптических кривых, определенных над простыми полями. В настоящей Рекомендации определяются новые структуры данных, версия которых изменена на v3 для поддержки реализации этих алгоритмов с асимметрическим спариванием. Для симметрического спаривания на суперсингулярных эллиптических кривых соответствующее поле в структурах данных ключей BB1 и BF остается неизменным, таким как в [IETF RFC 5091]. Еще три набора структур данных ключей определены для ECCSI, SM9 и SK-KEM. Соответственно:

```
BB1MasterSecret ::= SEQUENCE {
    version     INTEGER { v3(3) },
    alpha       INTEGER,
    beta        INTEGER,
    gamma       INTEGER
}
```

– для реализаций с асимметрическим спариванием альфа в пункте 9.3 [ISO/IEC 18033-5] принимает значение s1, бета – s2, а гамма – s3;

```
BB1PublicParameters ::= SEQUENCE {
    version     INTEGER { v3(3) },
    curve       OBJECT IDENTIFIER,
    hashfcn     OBJECT IDENTIFIER,
    pairing     PAIRING OPTIONAL,
    p           INTEGER OPTIONAL,
    q           [0] IMPLICIT INTEGER OPTIONAL,
    pointP      FpPoint,
    pointQ      [1] EXPLICIT FpxPoint OPTIONAL,
    pointP1     FpPoint,
    pointP2     [2] EXPLICIT FpxPoint OPTIONAL,
    pointP3     FpPoint,
    v           FpxElement
}
```

– Pairing (спаривание) определяет, какой тип билинейного отображения следует использовать для генерируемых параметров. Поддерживаются три типа спаривания: образование пар по Вейлю; образование пар по Тейту и оптимальное спаривание по Атэ.

– p и q становятся необязательными. Для некоторых типов кривых, таких как BN, BLS-12 и т. д., значения p и q предварительно определены идентификаторами объекта кривой (OID), а следовательно, нет необходимости указывать их снова.

– Для реализаций с асимметрическим спариванием pointP и pointQ в пункте 9.3 [ISO/IEC 18033-5] принимают значения Q1 в $G1$ и Q2 в $G2$. Для симметрического спаривания pointP равно pointQ, поэтому pointQ является необязательным.

– Для реализаций с асимметрическим спариванием pointP1 и pointP3 в пункте 9.3 [ISO/IEC 18033-5] принимают значения R и T.

– pointP2 для реализации с асимметрическим спариванием, таким как оптимальное спаривание по Атэ на кривых BN, принимает значение поля расширения F_p. pointP2 является необязательным, поскольку если задано значение v, то pointP2 для выполнения алгоритма BB1-KEM не требуется.

– v – это результат спаривания, который представляет собой элемент поля расширения F_p. Для реализации с асимметрическим спариванием, таким как оптимальное спаривание по Атэ на кривых BN, поле расширения принимает значение F_p^k, где k – степень вложения. В этом случае v в пункте 9.3 [ISO/IEC 18033-5] принимает значение J.

– Значения других полей остаются неизменными, такими как в [IETF RFC 5091].

```
PAIRING ::= ENUMERATED{
        weil        (1)     – образование пар по Вейлю,
        tate        (2)     – образование пар по Тейту,
        optimalAte  (3)     – оптимальное спаривание по Атэ
}
FpPoint ::= SEQUENCE{
        x   INTEGER,
        y   INTEGER
}
```

FpPoint определяет точку на эллиптической кривой над простым полем. Точка имеет две координаты, обозначаемые как координата *x* и координата *y*. Обе координаты принимают большие целые значения.

```
FpxPoint ::= CHOICE{
        fpPoint     [1] EXPLICIT FpPoint,
        fp2Point    [2] EXPLICIT Fp2Point,
        fp3Point    [3] EXPLICIT Fp3Point,
        fp4Point    [4] EXPLICIT Fp4Point
}
```

– Fp2Point определяет точку на эллиптической кривой над полем F_p^2. Каждая координата точки принимает значение элемента F_p^2.

– Fp3Point определяет точку на эллиптической кривой над полем F_p^3. Каждая координата точки принимает значение элемента F_p^3.

– Fp4Point определяет точку на эллиптической кривой над полем F_p^4. Каждая координата точки принимает значение элемента F_p^4.

```
Fp2Point ::= SEQUENCE{
        x   Fp2Element,
        y   Fp2Element
}
```

– Fp2Point определяет точку на эллиптической кривой над полем F_p^2. Точка имеет две координаты, обозначаемые как координата *x* и координата *y*. Обе координаты принимают значения из F_p^2.

```
Fp3Point ::= SEQUENCE{
        x   Fp3Element,
        y   Fp3Element
}
```

– Fp3Point определяет точку на эллиптической кривой над полем F_p^3. Обе координаты точки принимают значения из F_p^3.

```
Fp4Point ::= SEQUENCE{
        x   Fp4Element,
        y   Fp4Element
}
```

– Fp4Point определяет точку на эллиптической кривой над полем F_p^4. Обе координаты точки принимают значения из F_p^4.

Fp2Element ::= SEQUENCE{

a INTEGER,

b INTEGER

}

– Fp2Element определяет элемент поля F_p^2, который может быть представлен в виде a + bα, где α – неквадратный корень из F_p.

Fp3Element ::= SEQUENCE{

a INTEGER,

b INTEGER,

c INTEGER

}

– Fp3Element определяет элемент поля F_p^3, который может быть представлен в виде a + bβ + cβ^2, где β – некубический корень из F_p.

Fp4Element ::= SEQUENCE{

a Fp2Element,

b Fp2Element

}

– Fp4Element определяет элемент поля F_p^4, который может быть представлен как башня двух элементов F_p^2.

FpxElement ::= CHOICE{

fp2Elemt [1] EXPLICIT Fp2Element

– для реализации на основе суперсингулярной эллиптической кривой,

fp12Elemt [2] EXPLICIT Fp12Element

– с использованием представления башни F_p->F_p^2->F_p^6->F_p^12,

fp16Elemt [3] EXPLICIT Fp16Element

– с использованием представления башни F_p->F_p^2->F_p^4->F_p^8->F_p^16,

fp18Elemt [4] EXPLICIT Fp18Element

– с использованием представления башни F_p->F_p^3->F_p^6->F_p^18,

fp24Elemt [5] EXPLICIT Fp24Element

– с использованием представления башни F_p->F_p^2->F_p^6->F_p^12->F_p^24

}

– FpxElement определяет представление башни элемента в *G*3. Спаривание *e* отображает два входа из *G*1 и *G*2 в элемент в *G*3. Для обычно используемых кривых, подходящих для спаривания, элементы в *G*3, как правило, представляются методом башни. Для разных степеней вложения могут применяться разные представления башни. В настоящей Рекомендации определено широко используемое представление башни элементов поля с 12-й, 16-й, 18-й и 24-й степенями вложения.

Fp12Element ::= SEQUENCE{

a Fp6Element,

b Fp6Element

}

– Fp12Element определяет элемент F_p^12 с представлением башни 2x3x2, и его следует использовать в реализации с кривыми BN, или BLS-12, или BLS-24.

Fp6Element ::= SEQUENCE{

a Fp2Element,

b Fp2Element,

c Fp2Element

}

– Fp6Element определяет элемент F_p^6 с представлением башни 3x2, и его следует использовать в реализации с кривыми BN, или BLS-12, или BLS-24.

Fp16Element ::= SEQUENCE{

a Fp8Element,

b Fp8Element

}

– Fp16Element определяет элемент F_p^16 с представлением башни 2x2x2x2, и его следует использовать в реализации с кривыми KSS-16.

Fp8Element ::= SEQUENCE{

a Fp4Element,

b Fp4Element

}

– Fp8Element определяет элемент F_p^8 с представлением башни 2x2x2, и его следует использовать в реализации с кривыми KSS-16.

Fp18Element ::= SEQUENCE{

a Fp6bElement,

b Fp6bElement,

c Fp6bElement

}

– Fp18Element определяет элемент F_p^18 с представлением башни 3x2x3, и его следует использовать в реализации с кривыми KSS-18.

Fp6bElement ::= SEQUENCE{

a Fp3Element,

b Fp3Element

}

– Fp6bElement определяет элемент F_p^6 с представлением башни 2x3, и его следует использовать в реализации с кривыми KSS-18.

Fp24Element ::= SEQUENCE{

a Fp12Element,

b Fp12Element

}

– Fp24Element определяет элемент F_p^24 с представлением башни 2x2x3x2, и его следует использовать в реализации с кривыми BLS-24.

BB1PrivateKeyBlock ::= SEQUENCE {

version INTEGER { v3(3) },

pointD0 FpxPoint,

pointD1 FpxPoint

}

– Значения pointD0 и pointD1 остаются неизменными, такими как в [IETF RFC 5091], но если BB1-KEM реализован с асимметрическим спариванием, то они берутся из $G2$. В этом случае pointD0 и pointD1 в пункте 9.3 [ISO/IEC 18033-5] принимают значения соответственно dID0 и dID1.

```
BFMasterSecret ::= SEQUENCE {
        version         INTEGER { v3(3) },
        masterSecret    INTEGER
}
```

– Значение каждого поля остается неизменным, таким как в [IETF RFC 5091].

```
BFPublicParameters ::= SEQUENCE {
        version         INTEGER { v3(3) },
        curve           OBJECT IDENTIFIER,
        hashfcn         OBJECT IDENTIFIER,
        pairing         PAIRING OPTIONAL,
        p               INTEGER OPTIONAL,
        q               [0] IMPLICIT INTEGER OPTIONAL,
        pointP          FpxPoint,
        pointPpub       FpxPoint
}
```

– Значение каждого поля остается неизменным, таким как в [IETF RFC 5091], но если BF-IBE реализован с асимметрическим спариванием, то pointP и pointPpub берутся из $G2$. В этом случае pointP и pointPpub в пункте 8.2 [ISO/IEC 18033-5] принимают значения соответственно Q и R.

```
BFPrivateKeyBlock ::= SEQUENCE {
        version         INTEGER { v3(3) },
        privateKey      FpPoint
}
```

– Значение каждого поля остается неизменным, таким как в [IETF RFC 5091]. Для реализаций с асимметрическим спариванием privateKey в пункте 8.2 [ISO/IEC 18033-5] принимает значение skID.

```
ECCSIMasterSecret ::= SEQUENCE {
        version         INTEGER { v3(3) },
        masterSecret    INTEGER
}
```

– masterSecret в [IETF RFC 6507] принимает значение KSAK.

```
ECCSIPublicParameters ::= SEQUENCE {
        version         INTEGER { v2(2) },
        curve           OBJECT IDENTIFIER,
        hashfcn         OBJECT IDENTIFIER,
        pointP          FpPoint,
        pointPpub       FpPoint
}
```

– pointP в [IETF RFC 6507] принимает значение G.

– pointPpub в [IETF RFC 6507] принимает значение открытого ключа аутентификации KMS (KPAK).

```
ECCSIPrivateKeyBlock ::= SEQUENCE {
        version         INTEGER { v2(2) },
        ssk             INTEGER,
        pvt             OCTET STRING
}
```

– ssk и pvt в [IETF RFC 6507] принимают значение соответственно секретного ключа подписи (SSK) и открытого маркера проверки (PVT).

```
SKMasterSecret ::= SEQUENCE {
        version         INTEGER { v3(3) },
        masterSecret    INTEGER
}
```

– masterSecret принимает значение z_T в [IETF RFC 6508] и s в пункте 9.2 [ISO/IEC 18033-5].

```
SKPublicParameters ::= SEQUENCE {
        version         INTEGER { v3(3) },
        curve           OBJECT IDENTIFIER,
        hashfcn         OBJECT IDENTIFIER,
        pairing         PAIRING OPTIONAL,
        p               INTEGER OPTIONAL,
        q               [0] IMPLICIT INTEGER OPTIONAL,
        pointP1         FpPoint,
        pointP1pub      [1] EXPLICIT FpPoint OPTIONAL,
        pointP2         [2] EXPLICIT FpxPoint OPTIONAL,
        pointP2pub      [3] EXPLICIT FpxPoint OPTIONAL,
        v               [4] EXPLICIT FpxElement
}
```

– Для реализаций с симметрическим спариванием на суперсингулярных кривых p и q определены в [IETF RFC 5091]. Для реализаций с асимметрическим спариванием p и q предопределены используемой кривой и становятся необязательными.

– pointP1 принимает значение P в [IETF RFC 6508] и Q1 в *G*1 в пункте 9.2 [ISO/IEC 18033-5].

– pointP1pub принимает значение Z_T в [IETF RFC 6508] и R в пункте 9.2 [ISO/IEC 18033-5]. pointP1pub может быть не нужен для других алгоритмов, таких как алгоритмы подписи, основанные на функции генерирования Сакаи–Касахары (SK), так что это необязательный элемент.

– pointP2 в пункте 9.2 [ISO/IEC 18033-5] принимает значение Q2 в *G*2, если SK-KEM реализован с асимметрическим спариванием. pointP2 не нужен для выполнения SK-KEM, так что это необязательный элемент.

– pointP2pub принимает значение [*ib.msk*]Q2. Он не нужен для SK-KEM, но может потребоваться для других алгоритмов, таких как алгоритмы подписи, основанные на функции генерирования ключа SK, так что это необязательный элемент.

```
SKPrivateKeyBlock ::= SEQUENCE {
        version         INTEGER { v3(3) },
        privateKey      FpxPoint
```

}

– privateKey принимает значение RSK в [IETF RFC 6508] и skID в пункте 9.2 [ISO/IEC 18033-5].

```
SM9MasterSecret ::= SEQUENCE {
        version         INTEGER { v3(3) },
        masterSecret    INTEGER
}
```

– masterSecret принимает значение *ib.msk*, которое в пункте 7.4 [b-ISO/IEC 14888-3a] определяется значением U.

```
SM9PublicParameters ::= SEQUENCE {
        version         INTEGER { v3(3) },
        curve           OBJECT IDENTIFIER,
        hashfcn         OBJECT IDENTIFIER,
        pairing         PAIRING OPTIONAL,
        p               INTEGER OPTIONAL,
        q               [0] IMPLICIT INTEGER OPTIONAL,
        pointP1         FpPoint,
        pointP1pub      [1] EXPLICIT FpPoint OPTIONAL,
        pointP2         [2] EXPLICIT FpxPoint OPTIONAL,
        pointP2pub      [3] EXPLICIT FpxPoint OPTIONAL,
        v               [4] EXPLICIT FpxElement
}
```

– Для реализаций с симметрическим спариванием на суперсингулярных кривых p и q определены в [IETF RFC 5091]. Для реализаций с асимметрическим спариванием p и q предопределены используемой кривой.

– pointP1 принимает значение P в пункте 7.4 [ISO/IEC 14888-3].

– pointP1pub не нужен для SM9-IBS, но необходим для SM9-IBE, и в этом случае pointP2pub принимает значение [*ib.msk*]P.

– pointP2 в пункте 7.4 [ISO/IEC 14888-3] принимает значение Q. pointP2 не нужен для выполнения SM9-IBE, так что это необязательный элемент.

– pointP2pub в пункте 7.4 [ISO/IEC 14888-3a] принимает значение V. pointP2pub не нужен для выполнения SM9-IBE, так что это необязательный элемент.

```
SM9PrivateKeyBlock ::= SEQUENCE {
        version         INTEGER { v3(3) },
        privateKey      FpxPoint
}
```

– privateKey принимает значение X в пункте 7.4 [ISO/IEC 14888-3a] для подписи и значение *ib.prvk* в *G*1 для SM9-IBE и SM9-AKA.

Определение BFMasterSecret, BFPublicParameters и BFPrivateKeyBlock используется для алгоритмов с применением функции генерирования ключей Сакаи-Огиси-Касахары (SOK), таких как BF-IBE, Cha-Cheon-IBS, Hess-IBS, Fujioka-Suzuki-Ustaoglu-AKA, Smart-Chen-Cheng-AKA и Wang-AKA. Определение BB1MasterSecret, BB1PublicParameters и BB1PrivateKeyBlock используется для алгоритмов с применением функции генерирования ключей BB1, таких как BB1-KEM. SKMasterSecret, SKPublicParameters и SKPrivateKeyBlock используются для SK-KEM и, возможно, для других алгоритмов, основанных на функции генерирования ключей SK. SM9MasterSecret,

SM9PublicParameters и SM9PrivateKeyBlock используются для алгоритмов SM9, включая SM9-IBE, SM9-IBS и SM9-AKA. ECCSIMasterSecret, ECCSIPublicParameters и ECCSIPrivateKeyBlock используются для ECCSI.

Если необходимо обеспечить защиту закрытого ключа, то следует использовать структуру данных EncryptedPrivateKeyInfo, определенную в [IETF RFC 5958].

```
EncryptedPrivateKeyInfo ::= SEQUENCE {
        encryptionAlgorithm    EncryptionAlgorithmIdentifier,
        encryptedData          EncryptedData
}
EncryptionAlgorithmIdentifier ::= AlgorithmIdentifier
EncryptedData ::= OCTET STRING
AlgorithmIdentifier ::= SEQUENCE {
        algorithm              OBJECT IDENTIFIER,
        parameters             ANY DEFINED BY algorithm OPTIONAL
}
```

Приложение C

Операции управления ключами

(Данное Приложение является неотъемлемой частью настоящей Рекомендации)

В системе IBC к операциям управления ключами относятся инициализация системы, предоставление идентичности или закрытых ключей, аннулирование идентичности или закрытых ключей и публикация системных параметров. Инициализация системы включает шаг вызова функции **IBSetup**, а предоставление закрытых ключей – шаг вызова функции **IBExtract**. Для этих операций требуется взаимодействие между управляющим объектом и KMS. В настоящей Рекомендации для обмена сообщениями между этими двумя сторонами используется KMIP. Расширение, необходимое для удовлетворения новых требований поддерживаемых алгоритмов **IBSetup** и **IBExtract**, описано в Дополнении II. Протоколы взаимодействия между SecM и управляющими объектами для устройств IoT без eUICC определяются на основе HTTP. Для eUICC используются стандарты [b-GSMA SGP.02], которые при необходимости могут быть расширены.

C.1 Инициализация системы

В каждой системе IBC необходимо до предоставления KMS пользователям выполнить процесс инициализации системы. В этом процессе KMS выполняет одну или несколько функций **IBSetup** для генерирования одного или нескольких наборов пар ключей *ib.msk* и *ib.pubparam*. Метод повышения безопасности KMS выходит за рамки данной Рекомендации. Согласно передовой практике ключи *ib.msk* должны генерироваться и храниться в HSM. По возможности следует организовать распределенную схему генерирования ключей, в которой применяется схема распределения секретного кода с разделением *ib.msk* на фрагменты и распределением фрагментов секретного кода и функции генерирования закрытых ключей по нескольким KMS. В этом случае закрытый ключ, соответствующий идентификатору, может быть генерирован правильно только при надлежащем функционировании KMS, число которых превышает пороговое.

См. рисунок C.1.

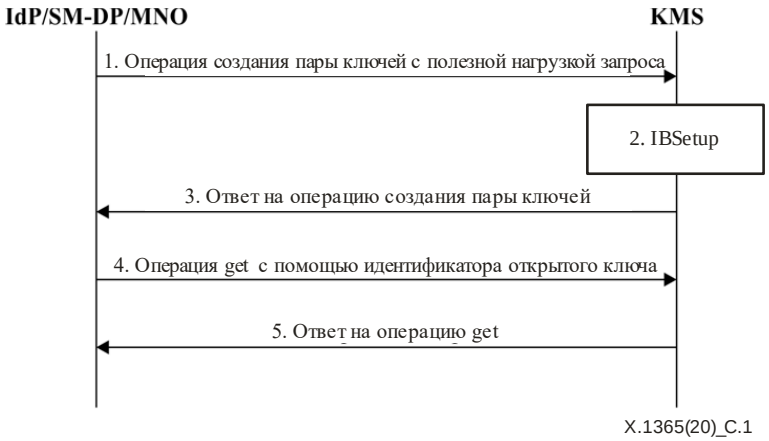

Рисунок C.1 – Инициализация системы с применением протокола взаимодействия для управления ключами

Начальные условия

Предполагается, что IdP/SM-DP/MNO играет роль инициатора системы и отвечает за процесс инициализации системы. Прежде чем IdP/SM-DP/MNO будет иметь возможность вызвать функцию **IBSetup** в KMS, должны быть выполнены следующие условия:

a) между IdP/SM-DP/MNO и KMS установлен защищенный канал;

b) IdP/SM-DP/MNO завершил(а) процесс аутентификации в KMS, и аутентифицированный(ая) IdP/SM-DP/MNO получил(а) разрешение (авторизацию) на выполнение запроса **IBSetup**.

Процедура

1) IdP/SM-DP/MNO подготавливает полезную нагрузку запроса и вызывает операцию создания пары ключей для передачи кодированного сообщения-запроса в KMS;

2) KMS проверяет правильность запроса и наличие у IdP/SM-DP/MNO разрешения на вызов этой операции. Если какое-либо из этих условий не выполнено, KMS возвращает ответ с отказом. В противном случае KMS выполняет функцию **IBSetup** с параметрами, указанными в запросе;

3) KMS возвращает IdP/SM-DP/MNO ответ об исполнении. Если операция выполнена успешно, KMS как минимум возвращает соответственно уникальный идентификатор закрытого ключа *ib.msk* и уникальный идентификатор открытого ключа *ib.pubparam*;

4) как вариант, если операция создания пары ключей выполнена успешно, IdP/SM-DP/MNO может вызвать операцию get с помощью уникального идентификатора открытого ключа, содержащегося в последнем ответе, чтобы получить открытые параметры *ib.pubparam*;

5) KMS возвращает значение ключа вновь сгенерированных открытых параметров.

Расширение KMIP для поддержки этой операции описано в Дополнении II

Условие завершения. KMS успешно инициализирована, и IdP/SM-DP/MNO располагает уникальным идентификатором закрытого ключа и уникальным идентификатором открытого ключа для доступа соответственно к сгенерированному MSK *ib.msk* и открытым параметрам *ib.pubparam*. IdP/SM-DP/MNO использует уникальный идентификатор закрытого ключа для вызова операции подписи, чтобы генерировать закрытые ключи на основе идентичности, и уникальный идентификатор открытого ключа для вызова операции get в целях получения открытых параметров.

C.2 Инициализация устройства

Операция инициализации устройства заключается в подготовке устройства к предоставлению идентичности и ключей. Для устройств IoT с eUICC и без eUICC выполняются разные процедуры инициализации устройства.

C.2.1 Случай 1. Инициализация eUICC

Для eUICC идентичность и соответствующий закрытый ключ *ib.prk* и открытые параметры *ib.sysparam* загружаются в профиль домена безопасности выдавшего органа (ISD). Следовательно, по завершении процесса инициализации устройства карта eUICC должна быть готова к созданию профиля ISD. В соответствии с [b-GSMA SGP.02] выполняется операция регистрации. Ниже воспроизводится пункт 3.5.1 [b-GSMA SGP.02].

- Регистрация eUICC в SM-SR

Начальное условие

a. eUICC изготовлены, и предоставляемый профиль загружен и активен в сети предоставляющего услуги оператора. eUICC протестированы и готовы к поставке. Каждая eUICC содержит соответствующий набор информации eUICC (EIS).

Процедура

1) Производитель eUICC (EUM) направляет в выбранную SM-SR запрос на регистрацию eUICC, содержащий EIS.

2) SM-SR сохраняет EIS в своей базе данных с идентификатором eUICC-ID (EID) в качестве параметра ключа.

3) SM-SR подтверждает EUM успешную регистрацию. Сообщение с подтверждением содержит EID.

Условие завершения. eUICC зарегистрирована в SM-SR и готова к загрузке профиля. Теперь она может быть отправлена производителю устройств межмашинной связи.

C.2.2 Случай 2. Инициализация устройства IoT без eUICC

Для устройств IoT без eUICC выполняется следующая операция регистрации.

- Регистрация SecM в AuC.

Начальное условие

a. SecM изготовлен, и устройство IoT в состоянии связаться с IdP в сети оператора.

Процедура

1) SecM посылает в AuC запрос на получение данных передачи для SecM;

2) AuC генерирует идентификатор передачи (PROV.ID) и соответствующее удостоверение аутентификации (PROV.CRED) запрашивающего SecM;

3) AuC направляет SecM PROV.ID и PROV.CRED. В том же сообщении AuC также направляет SecM идентификатор ключа IdP.ID и соответствующий открытый ключ IdP.PUK или *ib.sysparam*, если у SecM отсутствует возможность выполнения протокола TLS;

4) SecM надежно хранит PROV.ID и PROV.CRED, а также IdP.ID и IdP.PUK или *ib.sysparam*, если они предоставлены. SecM защищает IdP.ID и IdP.PUK или *ib.sysparam* от санкционированного изменения.

Условие завершения. SecM зарегистрирован в AuC и готов к предоставлению идентичности и ключей.

C.3 Поиск открытых параметров

Для получения открытых параметров системы IBC, с которыми он зарегистрирован, объект использует процедуру предоставления идентичности или ключей. Объект, который может быть устройством IoT или управляющим объектом в системе IBC, следуя спецификации, определенной в разделе 4 [IETF RFC 5408], извлекает из известного PPS открытые параметры другой системы IBC. Системные параметры IBESysParams в ответе в соответствии с [IETF RFC 5408] заменяются системными параметрами IBSysParams, определенными в настоящей Рекомендации. В [IETF RFC 5408] предполагается, что запрашивающее устройство IoT может установить защищенный канал на основе TLS с запрашиваемым PPS. Если такое требование не может быть выполнено, то в IBSysParams должны присутствовать действительные алгоритм подписи signatureAlgoritm и поле подписи. После получения IBSysParams выполняется надлежащий процесс проверки подписи, и только в том случае, если подпись в IBSysParams является действительной, а открытый ключ проверки подписи – подлинным и действительным, полученные открытые параметры принимаются.

C.4 Предоставление идентичности и ключей

Операция предоставления идентичности и ключей включает процедуру присвоения идентичности, извлечения закрытого ключа и распределения ключей. После процесса инициализации в устройствах имеются только предварительные идентификационные данные. IdP, SM-DP или MNO должны определить, какую идентичность следует присвоить запрашивающему устройству, а затем связаться с KMS, чтобы генерировать соответствующий закрытый ключ и, наконец, безопасно передать устройству идентификационные данные, закрытый ключ и открытые параметры.

См. рисунок C.2.

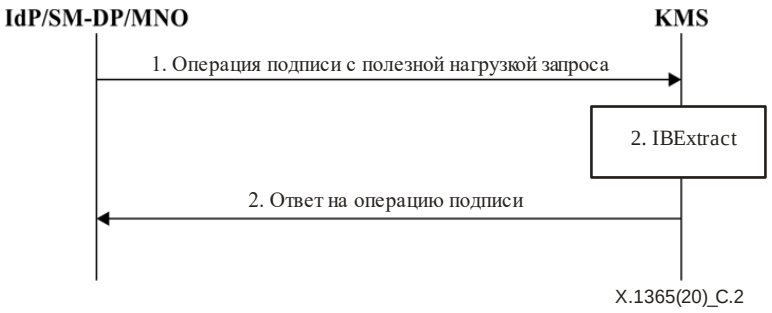

Рисунок C.2 – Генерирование закрытого ключа с применением протокола взаимодействия при управлении ключами

- **Генерирование закрытого ключа с применением KMIP**

Начальные условия

Предполагается, что IdP/SM-DP/MNO выполняет функцию генерирования закрытого ключа *ib.prk*. Прежде чем IdP/SM-DP/MNO будет иметь возможность вызвать функцию **IBExtract** в KMS, должны быть выполнены следующие условия:

a. между IdP/SM-DP/MNO и KMS установлен защищенный канал;

b. IdP/SM-DP/MNO завершил(а) процесс аутентификации в KMS, и аутентифицированный(ая) IdP/SM-DP/MNO получил(а) разрешение на выполнение запроса **IBExtract**.

Процедура

1) IdP/SM-DP/MNO подготавливает полезную нагрузку запроса и вызывает операцию подписи для передачи кодированного сообщения-запроса в KMS.

2) KMS проверяет правильность запроса и наличие у IdP/SM-DP/MNO разрешения на вызов этой операции. Если какое-либо из этих условий не выполнено, KMS возвращает ответ с отказом. В противном случае KMS выполняет функцию **IBExtract** с *ib.msk, ib.pubparam* и параметрами, указанными в запросе.

3) KMS возвращает IdP/SM-DP/MNO ответ об исполнении. Если операция прошла успешно, то KMS возвращает генерированный закрытый ключ *ib.prk* в форме IBPrivateKeyBlock, который представляет собой процедуру CHOICE, определенную в ASN.1, следующим образом:

```
IBPrivateKeyBlock ::= CHOICE{
        bb1PrivateKeyBlock      BB1PrivateKeyBlock,
        bfPrivateKeyBlock       BFPrivateKeyBlock,
        eccsiPrivateKeyBlock    ECCSIPrivateKeyBlock,
        skPrivateKeyBlock       SKPrivateKeyBlock,
        sm9PrivateKeyBlock      SM9PrivateKeyBlock
}
```

Расширение KMIP для поддержки этой операции описано в Дополнении II.

Условие завершения. IdP/SM-DP/MNO получил(а) закрытый ключ, соответствующий запрашиваемой идентичности.

- **Предоставление идентичности/ключей eUICC**

Начальные условия

a. eUICC зарегистрирована в SM-SR и готова к загрузке профиля.

b. SM-DP создала неперсонализированный профиль на основе описания профиля, предоставленного MNO.

c. У MNO имеется запрос на некоторое количество профилей eUICC.

d. Неперсонализированный профиль проверен на eUICC целевого типа с использованием процедуры проверки неперсонализированного профиля.

Процедура

1) MNO делает заказ профиля выбранной функции SM-DP. Детали процесса заказа профиля приведены в пункте 3.5.3 [b-GSMA SGP.02];

2) SM-DP создает персонализированный профиль с использованием данных, полученных от MNO. В частности, SM-DP использует в качестве идентификатора (идентичности) выбранный международный идентификатор абонента подвижной связи (IMSI) для выполнения операции подписи в KMS, как указано в процедуре генерирования закрытого ключа с применением KMIP, с тем чтобы генерировать закрытый ключ для выбранного IMSI. Генерированный закрытый ключ и ibPublicParameters из IBSysParams включаются в профиль в качестве ключей;

3) целевой профиль передается от MNO в eUICC. Детали процесса загрузки и установки профиля приведены в пункте 3.5.4 [b-GSMA SGP.02];

4) целевой профиль eUICC активизируется посредством SM-SR или SM-DP и SM-SR. Информация о конкретных шагах по активизации профиля см. в пункте 3.5.6 или пункте 3.5.7 [b-GSMA SGP.02].

Условие завершения. Целевой профиль активизирован в eUICC. Ранее активизированный профиль аннулирован. EIS находится в актуальном состоянии.

- **Предоставление идентичности и ключей для устройств IoT без eUICC**

Случай 1. SecM имеет возможность установления сеанса TLS с IdP

Начальное условие

a. SecM зарегистрирован в AuC.

Процедура

1) SecM устанавливает сеанс TLS с IdP и успешно проверяет действительность сертификата TLS IdP;
2) SecM выполняет процедуру веб-аутентификации у IdP, используя PROV.ID и PROV.CRED;
3) IdP выбирает идентичность, присвоенную запрашивающему устройству, и выполняет операцию подписи в KMS, как указано в процедуре генерирования закрытого ключа с применением KMIP, с тем чтобы генерировать закрытый ключ для выбранной идентичности;
4) IdP передает присвоенную идентичность, генерированный закрытый ключ и открытые параметры в SecM посредством сеанса TLS;
5) SecM надежно хранит закрытый ключ, а открытые параметры защищает от несанкционированного изменения.

Условие завершения. Целевой ключ предоставлен SecM.

Для завершения процедуры предоставления идентичности и ключей SecM и IdP должны следовать протоколу, определенному в разделе 5 и в [IETF RFC 5408]. В ответе структура данных IBEPrivateKeyReply, определенная в [IETF RFC 5408], заменяется на IBPrivateKeyReply.

```
IBPrivateKeyReply ::= SEQUENCE SIZE (1..MAX) OF IBPrivateKey

IBPrivateKey ::= SEQUENCE {
        pkgIdentity         IBIdentityInfo OPTIONAL,
        pkgAlgorithm        OBJECT IDENTIFIER,
        pkgKeyData          IBPrivateKeyBlock – определен алгоритм pkgAlgorithm,
        pkgOptions          SEQUENCE SIZE (1..MAX) OF PKGOption,
        ibSysParams         IBSysParams OPTIONAL
}

PKGOption ::=  SEQUENCE  {
        optionID        OBJECT IDENTIFIER,
        optionValue     OCTET STRING
}
```

Случай 2. В SecM отсутствует реализация TLS

Начальное условие

a. SecM зарегистрирован в AuC.

Процедура

1) SecM генерирует ключ шифрования ключей (KEK) и кодирует запрос на предоставление ключа (IBKeyProvRequest). Этот запрос включает в себя KEK, идентификатор передачи (PROV.ID) и удостоверение (PROV.CRED), зашифрованные с использованием открытого

ключа IdP с идентификатором IdP.ID. Результат шифрования кодируется как EncryptedMsg. Зашифрованный запрос передается в IdP как тело запроса HTTP POST;

2) IdP расшифровывает зашифрованный текст, используя закрытый ключ с идентификатором IdP.ID из запроса, и проверяет актуальность временной метки, правильность значения счетчика или и то и другое. Если запрос не проходит эти проверки, IdP возвращает ответ, указывающий на ошибку. IdP также проверяет в AuC правильность PROV.ID и PROV.CRED. Если эта проверка не проходит, IdP возвращает ответ, указывающий на ошибку. IdP выбирает идентичность, присвоенную запрашивающему устройству, и выполняет операцию подписи в KMS, как указано в процедуре генерирования закрытого ключа с применением KMIP, с тем чтобы генерировать закрытый ключ для выбранной идентичности;

3) IdP шифрует генерированный закрытый ключ и при необходимости идентификационные данные и открытые параметры, закодированные как IBKeyProvisionData, с помощью ключа шифрования ключей (KEK) с использованием определенного алгоритма, переданного в запросе (keyProtAlg). Зашифрованный текст кодируется как EncryptedMsg. IdP передает зашифрованный ответ SecM как тело ответа HTTP;

4) SecM расшифровывает ответ и получает присвоенную идентичность, закрытый ключ и открытые параметры. SecM надежно хранит закрытый ключ, а открытые параметры защищает от несанкционированного изменения.

Условие завершения. Целевой ключ предоставлен SecM.

```
IBKeyProvisionRequest ::= SEQUENCE{
        version         INTEGER { v1(1) },
        timer           Time OPTIONAL,
        counter         INTEGER         OPTIONAL,
        identity        OCTET STRING,
        credential      OCTET STRING,
        keyProtAlg      OBJECT IDENTIFIER,
        kek             OCTET STRING
}
Time ::= CHOICE {
    utcTime         UTCTime,
    generalTime     GeneralizedTime
}
IBKeyProvisionResponse ::= SEQUENCE SIZE(1..MAX) OF IBKeyProvisionData
IBKeyProvisionData ::= SEQUENCE{
        identity        OCTET STRING OPTIONAL,
        ibSysParams     IBSysParams OPTIONAL,
        ibPrivateKey    IBPrivateKeyBlock
}
EncryptedMsg ::= SEQUENCE {
        encryptionAlgorithm   EncryptionAlgorithmIdentifier,
        encryptedData         EncryptedData
}
EncryptionAlgorithmIdentifier ::= AlgorithmIdentifier
```

EncryptedData ::= OCTET STRING

C.5 Аннулирование идентичности и ключей

Если в системе IBC по тем или иным причинам нужно аннулировать идентичность, например потому, что ее владелец отменил подписку на услугу или соответствующий закрытый ключ был взломан, то идентичность отзывается, и по соображениям безопасности может потребоваться уничтожение соответствующего закрытого ключа. Если идентичность аннулирована, ей присваивается статус аннулированной. В случае если запись запрашивает статус аннулированной идентичности, IdP/SM-DP/MNO направляют в ответ правильное значение, определенное в OISP. Для более эффективной проверки статуса идентичности можно регулярно извлекать IRL из IdP/SM-DP/MNO и сохранять его локально, а затем проверять по новому IRL, аннулирована ли идентичность, не запрашивая информацию о статусе в онлайновом режиме для каждой идентичности. Для eUICC процесс уничтожения закрытого ключа можно реализовать, сначала отключив профиль, а затем удалив его из eUICC.

- **Аннулирование идентичности и ключей для eUICC**

Начальное условие

a. Целевой профиль активизирован в eUICC.

Процедура

1) MNO запускает отключение профиля посредством процесса SM-DP. Детали процесса отключения профиля приведены в пункте 3.5.8 [b-GSMA SGP.02]. SM-DP присваивает идентичности статус аннулированной;

2) MNO запускает процесс удаления профиля. Информацию о конкретных шагах по удалению ISD-P см. в пункте 3.5.10 [b-GSMA SGP.02]. SM-DP переводит идентичность в статус аннулированной, и если процесс удаления ISD-P успешно завершен, то идентичность также переводится в статус удаленной. Когда объект запрашивает статус идентичности, SM-DP отвечает соответствующим образом согласно записи статуса. SM-DP периодически публикует список статусов идентичностей, аннулированных за прошедший период.

Условие завершения. Целевой профиль отключен и удален из eUICC.

- **Аннулирование идентичности и ключей для устройств IoT без eUICC**

Если идентичность аннулирована, IdP присваивает ей статус аннулированной. Когда объект запрашивает статус идентичности, IdP отвечает соответствующим образом согласно записи статуса. IdP периодически публикует список статусов идентичностей, аннулированных за прошедший период.

Описание процесса запуска аннулирования и сохранения статуса идентичности выходят за рамки данной Рекомендации.

- **Онлайновый протокол статуса идентичности**

При большом количестве устройств IoT, подключающихся к сети оператора электросвязи, может потребоваться, чтобы SM-DP, IdP или устройство IoT своевременно получали информацию о статусе в отношении аннулированных идентичностей устройств IoT. В настоящей Рекомендации определен протокол OISP, позволяющий SM-DP, IdP или устройству IoT определять текущий статус идентичности с помощью онлайновых запросов. Клиент OISP отправляет запрос статуса ответчику OISP и приостанавливает прием соответствующих идентичностей до получения ответа. OISP имеет сходство с онлайновым протоколом статуса сертификата (OCSP) [IETF RFC 6960].

Запрос OISP содержит следующие данные:

```
OISPRequest ::= SEQUENCE {
    version     INTEGER { v1(1) },
    identity    IBIdentityInfoSet
}
```

– version указывает версию протокола, которой для этого документа является v1(1).

– identity – запрос OISP.

```
IBIdentityInfoSet ::= SEQUENCE SIZE(1..MAX) OF IBIdentityInfo

IBIdentityInfo ::= SEQUENCE {
        domainName   IA5String OPTIONAL,
        domainSerial   INTEGER OPTIONAL,
        identityType   OBJECT IDENTIFIER OPTIONAL,
        identityData   OCTET STRING
}
```

– domainName – необязательное поле, а IA5String представляет собой URI [b-URI] или IRI [b-IRI].

– domainSerial – необязательное поле, содержащее целое число, определяющее уникальный набор открытых параметров IBC в том случае, если в одном домене используется более одного набора параметров.

– identityType – необязательное поле, содержащее идентификатор объекта, определяющий формат, в котором кодируется поле identityData. Если это поле отсутствует, используется тип идентичности по умолчанию.

– identityData – данные целевой идентичности.

По получении запроса ответчик OISP проверяет, правильно ли сформировано сообщение и содержит ли запрос информацию, необходимую ответчику. Если проверка оказалась неудачной, ответчик OISP выдает сообщение об ошибке; в противном случае он возвращает точный ответ в соответствии со статусом запрашиваемых идентичностей:

```
OISPResponse ::= SEQUENCE {
        responseStatus          OISPResponseStatus,
        responseData            OISPResponseData OPTIONAL
}
```

– responseStatus указывает статус обработки предыдущего запроса.

– responseData – необязательное поле, содержащее данные ответа на запрос. Если значение responseStatus одно из состояний ошибки, то поле responseData не устанавливается.

```
OISPResponseStatus ::= ENUMERATED {
      successful   (0)   – ответ имеет действительные подтверждения,
      malformedRequest      (1)   – недопустимый запрос подтверждения,
      internalError    (2)   – внутренняя ошибка отправителя,
      tryLater         (3)   – попробуйте позже,
                       (4)    – не используется,
      unauthorized     (5)   – запрос не авторизован
}

OISPResponseData ::= SEQUENCE {
      version                 INTEGER { v1(1) },
      producedAt              GeneralizedTime,
      hashAlgorithm           AlgorithmIdentifier OPTIONAL,
      tbsIdStatus             SEQUENCE OF SingleIdStatus,
      signatureAlgorithm      AlgorithmIdentifier OPTIONAL,
      signature               BIT STRING OPTIONAL,
```

```
    certs                [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL
}
```

– Для данной версии основного синтаксиса ответов значение version должно быть v1(1).

– producedAt – момент времени, в который ответчик OISP подписал данный ответ.

– hashAlgorithm определяет алгоритм хеширования, по которому генерируется значение idHash в поле tbsIdStatus, если таковое имеется. Это поле является необязательным и его значением по умолчанию служит идентификатор объекта SHA256 без параметров.

– tbsldStatus указывает ответы для каждой запрашиваемой идентичности.

– signatureAlgorithm – необязательное поле, указывающее алгоритм, который использовался для подписания ответа.

– signature (подпись) вычисляется как результат применения DER ASN.1 от поля producedAt до tbsIdStatus с указанным алгоритмом подписи. Это поле является необязательным и может не устанавливаться, если у клиента OISP имеются другие методы, позволяющие гарантировать подлинность ответа. Например, ответ передается по защищенному каналу TLS между клиентом и ответчиком.

– certs – необязательное поле, указывающее сертификат, который помогает клиенту OISP проверить подпись ответчика. Структура сертификата определена в [IETF RFC 5280].

```
SingleIdStatus ::= SEQUENCE {
    idHash           OCTET STRING OPTIONAL,
    identityID       IBIdentityInfo OPTIONAL,
    identityStatus   IdentityStatus
}
```

– idHash – это необязательное поле, содержащее хеш запрашиваемой идентичности. Если поле identityID слишком длинное, то для представления запрашиваемой идентичности может использоваться idHash. identityID – необязательное поле, содержащее поле IBIdentityInfo целевой идентичности из запроса.

– identityStatus указывает статус идентичности в предыдущем запросе.

```
IdentityStatus ::= CHOICE {
    good                 [0]   IMPLICIT NULL,
    revoked              [1]   IMPLICIT RevokedInfo,
    unknown              [2]   IMPLICIT UnknownInfo,
    updated              [3]   IMPLICIT IBIdentityInfo,
    revokedAndDeleted    [4]   IMPLICIT RevokedInfo
}
UnknownInfo ::= NULL
```

– Состояние good указывает на положительный ответ на запрос статуса.

– Состояние revoked указывает, что идентичность аннулирована либо временно, либо навсегда, и значением является информация об аннулировании.

– Состояние unknown означает, что ответчику неизвестна информация о запрашиваемом сертификате.

– Состояние updated указывает, что идентичность обновлена, и значением является вновь присвоенная идентичность для запрашиваемой идентичности.

– Состояние revokedAndDeleted указывает, что идентичность аннулирована, а закрытый ключ уничтожен на удаленном устройстве.

```
RevokedInfo ::= SEQUENCE {
```

```
    revocationTime      GeneralizedTime,
    revocationReason    [0] EXPLICIT IRLReason OPTIONAL
}
IRLReason ::= ENUMERATED {
        unspecified             (0),
        keyCompromise           (1),
        pkgCompromise           (2),
        affiliationChanged      (3),
        superseded              (4),
        cessationOfOperation    (5),
        identityHold            (6),
                                (7) не используется,
        removeFromIRL           (8),
        privilegeWithdrawn      (9)
}
```

- **Список аннулированных идентичностей**

Помимо использования OSIP для ответа на запросы статуса идентичности, такой объект, как IdP или SM-DP, может регулярно публиковать полный список аннулированных идентичностей – IRL. Для ускорения процесса проверки статуса идентичности проверяющий статус объект с хранилищем большой емкости может запросить IRL и хранить его локально. Проверяющий объект может определять по IRL, приемлема ли идентичность для определенных операций, таких как авторизация доступа к сети. Если данная идентичность отсутствует в IRL, то предполагается, что она действительна. Чтобы повысить эффективность системы, IdP/SM-DP/MNO может публиковать только сведения об идентичностях, аннулированных с определенного момента. Этот список называется дельта-IRL. Он содержит информацию об идентичностях, аннулированных с момента публикации полного IRL. Использование списка дельта-IRL может значительно сократить накладные расходы на связь и время обработки IRL. IRL аналогичен списку аннулированных сертификатов (CRL) [IETF RFC 5280].

IRL определяется следующим образом:

```
IdentityRevocationList ::= SEQUENCE {
    tbsIdentityList         TBSIdentityRevocationList,
    signatureAlgorithm      AlgorithmIdentifier OPTIONAL,
    signatureValue          BIT STRING OPTIONAL
}
```

– tbsIdentityList – список аннулированных идентичностей с дополнительной информацией, такой как время аннулирования.

– signatureAlgorithm определяет алгоритм, используемый издателем IRL для подписи списка. Это поле является необязательным и отсутствует, если нет поля signatureValue.

– signatureValue определяет значение подписи, генерированной издателем в tbsIdentityList. Это поле является необязательным и отсутствует, если запрашивающий клиент располагает другими средствами гарантирования подлинности полученного списка.

```
TBSIdentityRevocationList ::= SEQUENCE {
    version                 INTEGER { v1(1) },
    issuer                  Name,
```

```
    irlNumber               INTEGER OPTIONAL,
    deltaList               BOOLEAN OPTIONAL,
    thisUpdate              Time,
    nextUpdate              Time OPTIONAL,
    domainName              IA5String OPTIONAL,
    domainSerial            INTEGER OPTIONAL,
    revokedIdentities       SEQUENCE OF SEQUENCE  {
        identity            IBIdentityInfo,
        revocationDate      Time,
        irlEntryExtensions  Extensions OPTIONAL
    } OPTIONAL,
    irlExtensions           [0] EXPLICIT Extensions OPTIONAL
}
Name ::= CHOICE { – импортируется из [IETF RFC 5280]
    rdnSequence  RDNSequence
}
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
RelativeDistinguishedName ::= SET SIZE (1..MAX) OF AttributeTypeAndValue
AttributeTypeAndValue ::= SEQUENCE {
    type     AttributeType,
    value    AttributeValue
}
AttributeType ::= OBJECT IDENTIFIER
AttributeValue ::= ANY – определяется параметром AttributeType
Extensions  ::=    SEQUENCE SIZE (1..MAX) OF Extension
Extension ::=SEQUENCE  {– импортируется из [IETF RFC 5280]
   extnID     OBJECT IDENTIFIER,
   critical   BOOLEAN DEFAULT FALSE,
   extnValue OCTET STRING
          – содержит кодировку DER значения ASN.1
          – соответствует типу расширения,
          – указанному в extnID
}
```

– version указывает версию структуры IRL.

– issuer – наименование организации, издавшей IRL.

– irlNumber – присвоенный издателем номер текущего IRL. Начинается с 0. Для каждой публикации полного IRL номер увеличивается на 1. Это необязательное поле.

– deltaList указывает, является ли текущий IRL списком дельта-IRL. Дельта-IRL содержит только информацию об идентичностях, аннулированных с момента публикации полного IRL, номер которого указан в поле irlNumber.

– thisUpdate указывает время создания данного IRL.
– nextUpdate определяет время создания следующего IRL. Это необязательное поле.
– domainName определяет домен идентичности IBC.
– domainSerial определяет номер домена идентичности IBC.
– revokedIdentities – набор аннулированных идентичностей.
  - identity – данные аннулированной идентичности.
  - revocationDate – время аннулирования идентичности.
  - irlEntryExtensions определяет возможные расширения параметра revokedIdentity. В настоящее время расширения не определены.
– irlExtensions определяет возможные расширения IRL. В настоящее время расширения не определены.

Приложение D

Аутентификация

(Данное Приложение является неотъемлемой частью настоящей Рекомендации)

В данном Приложении четыре существующих протокола аутентификации расширены для поддержки IBC.

D.1 Однопроходный протокол передачи секретных ключей

Данный протокол соответствует механизму передачи секретных ключей 2, описанному в [ISO/IEC 11770-3]. Он передает секретный ключ, генерированный, зашифрованный и подписанный объектом A, от объекта A к объекту B с явной аутентификацией ключа объектом A для объекта B и неявной аутентификацией ключа объектом B для объекта A. Явная аутентификация ключа объектом A для объекта B достигается тем, что объект A подписывает зашифрованный секретный ключ и зависящий от времени параметр (TVP). Неявная аутентификация ключа объектом B для объекта A достигается путем шифрования секретного ключа с помощью идентификатора объекта B, а это означает, что восстановить секретный ключ может только объект B. См. рисунок D.1.

Для выполнения этого протокола должны соблюдаться следующие требования.

– Объект A имеет закрытый ключ подписи *A.ib.prk*, соответствующий его идентификатору, и связанные с ним открытые параметры *A.ib.pubparam*.

– Объект B имеет закрытый ключ дешифрования *B.ib.prk*, соответствующий его идентификатору, и связанные с ним открытые параметры *B.ib.pubparam*.

– Объект A имеет доступ к аутентифицированной копии открытых параметров объекта B для шифрования *B.ib.pubparam* и идентификатора объекта B.

– Объект B имеет доступ к аутентифицированной копии открытых параметров объекта A для подписи *A.ib.pubparam* и идентификатора объекта A.

– Необязательный TVP может быть временной меткой или порядковым номером. Если используются метки времени, то объекты A и B осуществляют синхронизацию или используют доверенные временные метки третьей стороны.

– Объекты A и B могут иметь одни и те же открытые параметры, то есть *A.ib.pubparam* = *B.ib.pubparam.*

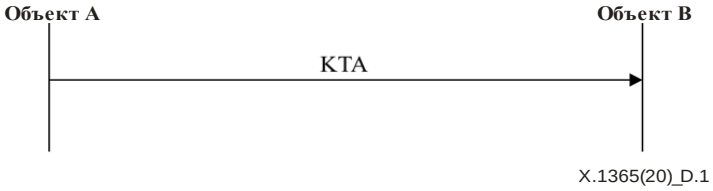

Рисунок D.1 – Однопроходный протокол передачи секретных ключей

1) Объект A генерирует случайный секретный ключ *K* необходимой длины.

2) Объект A генерирует *BE* = **IBEnc** (*B.ib.pubparam, ID_B,* [*ID_A*] || *K* || *Text1*). Поле Text1 может быть пустым, а поле *ID_A* является необязательным, если объект B располагает другими средствами получения идентификатора объекта A.

3) Объект A генерирует *S* = **IBSign** (*A.ib.pubparam, ID_A, A.ib.prk,* [*ID_B*] || TVP || BE || *Text2*). Поле Text2 может быть пустым, а поле *ID_B* является необязательным, если объекту B известен идентификатор, используемый *ID_B* для шифрования.

4) Объект A генерирует маркер KTA = [*ID_B*] || TVP || BE || *Text2* || S ||*Text3.*

5) Если TVP является временной меткой, то объект B проверяет, находится ли TVP в допустимых пределах разницы во времени. В противоположном случае объект B отклоняет маркер.

6) Если объект B может получить *ID_A* другими способами, а TVP является порядковым номером, то объект B сначала проверяет, больше ли порядковый номер, чем тот, что хранится в объекте B. В противоположном случае объект B отклоняет маркер.

7) Если объект B может получить *ID_A* другими способами, то объект B проверяет подпись *S* в KTA по **IBVerify**(*A.ib.pubparam, ID_A,* [*ID_B*] ||TVP || BE || *Text2,* S). Если подпись недействительна, то объект B отклоняет маркер.

8) Объект B расшифровывает *BE* с помощью [*ID_A*] || *K* || *Text1* = **IBDec**(*B.ib.pubparam, ID_B, B.ib.prk,* BE).

9) В случае если объект B может получить *ID_A* только после шага 8, то объект B проверяет актуальность TVP, если TVP является порядковым номером. Если TVP не актуален, объект B отклоняет маркер. Объект B проверяет подпись *S*. Если подпись недействительна, объект B отклоняет маркер.

10) Если все проверки прошли успешно, объект A и объект B используют *K* для защиты последующих сообщений. Оба объекта могут использовать функцию выработки ключей (KDF) [b-IEEE 1363] для генерирования ключей шифрования и аутентификации сообщений.

ПРИМЕЧАНИЕ 1. – Этот протокол может быть преобразован в односторонний протокол аутентификации объектов путем исключения BE из сообщения, подписанного объектом A и KTA. Это изменение приводит к однопроходной схеме аутентификации объекта, определенной в [b-ISO/IEC 9798-3].

ПРИМЕЧАНИЕ 2. – Этот протокол можно преобразовать в двусторонний протокол аутентификации объектов, потребовав, чтобы объект B возвращал *K* объекту A. Объект B аутентифицируется объектом A, демонстрируя свою способность восстановить *K*, для чего требуется закрытый ключ *B.ib.prk*.

ПРИМЕЧАНИЕ 3. – Для повышения эффективности можно использовать алгоритмы параллельного шифрования и подписи (signscription) на основе идентичности, такие как BLMQ [b-Barreto] и алгоритм шифрования подписи Чэня–Малоне–Ли [b-Chen].

D.2 TLS-IBS

В данном пункте описывается другой протокол аутентификации, именуемый TLS-IBS. Предполагается, что как на стороне сервера, так и на стороне устройства IoT имеются основанные на идентичности удостоверения, содержащие идентификационные данные, закрытый ключ подписи и открытые параметры KMS (например, KPAK, определенный в [IETF RFC 6507], в качестве параметра вычисления). Определения структуры открытых параметров KMS для поддерживаемых алгоритмов приведены в Приложении B.

Алгоритм TLS-IBS разработан на основе [IETF RFC 7250]. Традиционно клиент и сервер TLS обмениваются открытыми ключами, подтвержденными сертификатами инфраструктуры открытых ключей (PKI). Считается, что это сложно и может привести к недостаточной защите при использовании сертификатов PKI. Для упрощения обмена сертификатами в TLS предлагается использовать исходный открытый ключ, определенный в [IETF RFC 7250]. Другими словами, вместо передачи в сообщениях TLS полных сертификатов клиент и сервер обмениваются только открытыми ключами. Однако для привязки открытого ключа к идентичности предполагается использование внеполосного механизма. Для сетей IoT использование TLS с исходным открытым ключом особенно привлекательно, но связывание идентичностей с открытыми ключами может оказаться сложной задачей. Необходимость ведения большой таблицы соответствия идентичностей и открытых ключей на стороне сервера влечет за собой дополнительные расходы на техническое обслуживание, например устройства должны предварительно регистрироваться на сервере. Лучшим способом упростить привязку открытых ключей к предоставляющим их объектам может стать использование для аутентификации метода IBC, такого как открытый ключ ECCSI, описанный в [IETF RFC 6507]. В отличие от сертификатов МСЭ-T X.509 и исходных открытых ключей, открытый ключ IBC принимает форму идентичности объекта. Это помогает исключить необходимость привязки открытого ключа к предоставляющему его объекту.

При использовании IBS в качестве исходного открытого ключа для TLS во время установления соединения согласовываются алгоритмы подписи и хеширования. Установление соединения между

клиентом и сервером TLS осуществляется в соответствии с процедурами, определенными в [IETF RFC 7250] и TLS 1.3 [IETF RFC 8446], но с поддержкой алгоритмов IBS в качестве схем подписи.

Ниже описывается протокол TLS-IBS, разработанный на основе [IETF RFC 7250] и TLS 1.3 с использованием в качестве алгоритмов подписи ECCSI [IETF RFC 6507], IBS1 (Hess-IBS), IBS1 (Cha-Cheon-IBS) и SM9-IBS [ISO/IEC 14888-3]:

1) устройство IoT передает на сервер сообщение ClientHello с расширениями key_share, signature_algorithms, server_certificate_type и client_certificate_type, указывая, что оно поддерживает исходный открытый ключ и алгоритмы IBS;

2) сервер отправляет устройству IoT сообщение ServerHello с расширениями key_share, server_certificate_type, client_certificate_type и полями Certificate, CertificateRequest, CertificateVerify и Finished, указывающее, что исходный открытый ключ поддерживается, и включает свой идентификатор (ServerID) и параметры KMS (OID, KMS parameters) в часть сертификата. Структуры данных параметров KMS определены в пункте D.2.3. В сообщение CertificateVerify включается подпись, генерированная с помощью закрытого ключа сервера;

3) после проверки идентичности и подписи сервера устройство IoT отправляет на сервер свой исходный открытый ключ в полях Certificate, CertificateVerify и Finished. Устройство IoT включает в область сертификата свой идентификатор (ClientID) и параметры KMS (OID, KMS parameters), которые составляют исходный открытый ключ клиента. Структуры данных параметров KMS определены в пункте D.2.3. Также включается подпись, генерированная с помощью закрытого ключа клиента;

4) остальные шаги те же, что и для TLS 1.3 в [IETF RFC 8446].

См. рисунок D.2.

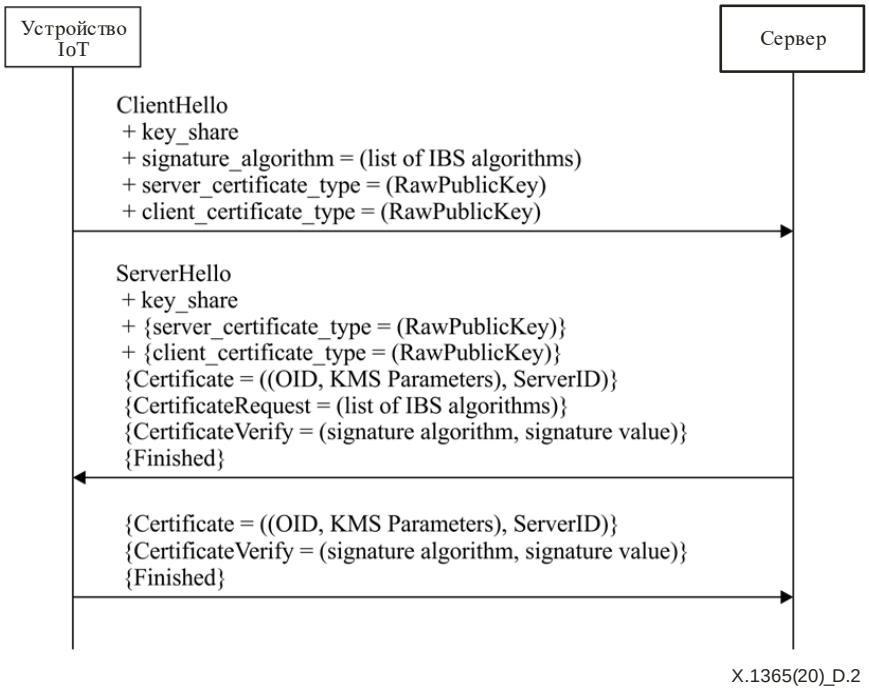

Рисунок D.2 – TLS-IBS

D.2.1 Сообщение ClientHello

Сообщение ClientHello имеет такой же формат, какой указан в TLS 1.3 [IETF RFC 8446], но для IBS необходимо расширить значения в алгоритме подписи.

Сообщение ClientHello указывает серверу типы сертификатов или исходных открытых ключей, поддерживаемых клиентом, а также типы сертификатов, которые клиент ожидает получить от сервера. Это сообщение включает требуемые алгоритмы IBS на основе порядка предпочтений клиента. В TLS

1.3 для алгоритмов подписи определена структура данных с именем SignatureScheme. Для поддержки алгоритма IBS ее необходимо расширить следующим образом:

```
enum {
    ...
  /* Алгоритм подписи IBS */
    eccsi_sha256 (0x0704),
    ibs1_sha256(0x0705)
    ibs2_sha256(0x0706)
    sm9_ibs_sm3(0x0707)
  /* Зарезервированные кодовые точки */
  private_use (0xFE00..0xFFFF),
  (0xFFFF)
} SignatureScheme;
```

Подробная информация о кодовых точках для расширенных алгоритмов подписи приведена в регистре TLS [b-IANA TLS REG].

D.2.2 Сообщение ServerHello

Сообщение ServerHello имеет такой же формат, какой указан в TLS 1.3 [IETF RFC 8446]. SignatureScheme расширяется так же, как и Client_Hello.

D.2.3 Сертификат сервера

Для сертификата сервера структура сертификата определяется как RawPublicKey в [b-IETF RFC 7250]. Как и в [IETF RFC 7250], для указания исходного открытого ключа и его криптографического алгоритма используется структура данных subjectPublicKeyInfo. В структуре subjectPublicKeyInfo определены два поля – algorithm и parameters. Поле algorithm определяет криптографический алгоритм, используемый с исходным открытым ключом, который представлен идентификаторами объекта; поле parameters содержит необходимые параметры, связанные с алгоритмом. Идентичность сервера должна содержаться в части subjectPublicKey.

ПРИМЕЧАНИЕ. – Идентичность должна соответствовать формату, определенному в Дополнении I.

```
subjectPublicKeyInfo ::= SEQUENCE {
        algorithm                   AlgorithmIdentifier,
        subjectPublicKey                BIT STRING
}
AlgorithmIdentifier ::= SEQUENCE {
        algorithm           OBJECT IDENTIFIER,
        parameters          ANY DEFINED BY algorithm OPTIONAL
}
```

При применении алгоритма IBS идентичность используется в качестве исходного открытого ключа, который можно преобразовать в строку OCTET. Следовательно, структура сертификата и subjectPublicKey могут многократно использоваться без изменений.

Поле algorithm в структуре AlgorithmIdentifier служит идентификатором объекта используемого алгоритма IBS. Кроме того, необходимо сообщить узлу-партнеру набор открытых параметров, используемых подписывающей стороной. Эта информация может содержаться в полезной нагрузке поля parameters в AlgorithmIdentifier. В соответствии с приведенными выше алгоритмами структурами открытых параметров являются соответственно ECCSIPublicParameters, BFPublicParameters и SM9PublicParameters, определенные в Приложении B.

Для поддержки алгоритмов IBS по протоколу TLS для генерирования сообщения CeritificateVerify необходимо определить структуру данных значения подписи.

- Структура данных ECCSI определяется следующим образом (на основе [IETF RFC 6507]):
  ECCSI-Sig-Value ::= SEQUENCE {
  r INTEGER,
  s INTEGER,
  pvt OCTET STRING
  }
  где pvt (как PVT, определенный в [IETF RFC 6507]) кодируется как 0x04 || координата x [v]G || координата y [v]G.
- Структура данных IBS1 определяется следующим образом:
  IBS1-Sig-Value ::= SEQUENCE {
  r INTEGER,
  s ECPoint
  }
  ECPoint ::= OCTET STRING, как определено в [IETF RFC 5480].
- Структура данных IBS2 определяется следующим образом:
  IBS2-Sig-Value ::= SEQUENCE {
  r ECPoint,
  s ECPoint
  }
- Структура данных SM9-IBS определяется следующим образом:
  SM9-Sig-Value ::= SEQUENCE {
  r INTEGER,
  s ECPoint
  }

Для того чтобы использовать алгоритм подписи с TLS, необходимо указать OID алгоритма подписи. В таблице D.1 приведена основная информация, необходимая для использования алгоритмов подписи IBS для TLS.

Таблица D.1 – Алгоритмы подписи на основе идентичности

Тип ключа	Документ	OID
ISO/IEC 14888-3 ibs-1	ISO/IEC 14888-3: механизм IBS1	1.0.14888.3.0.7
ISO/IEC 14888-3 ibs-2	ISO/IEC 14888-3: механизм IBS2	1.0.14888.3.0.8
SM9-IBS	ISO/IEC 14888-3: механизм китайской IBS	1.2.156.10197.1.302.1
Подписи без сертификатов на основе эллиптических кривых для основанного на идентичности шифрования (ECCSI)	Пункт 5.2 [IETF RFC 6507]	1.3.6.1.5.5.7.6.29

D.2.4 Сертификат клиента

Для поддержки IBS сертификат клиента расширяется таким же образом, как и сертификат сервера.

D.3 EAP-TLS-IBS

В данном пункте описывается протокол аутентификации EAP-TLS, расширенный для поддержки IBS. Как на стороне сети, так и на стороне UE имеются удостоверения на основе идентичности, содержащие идентификационные данные, закрытый ключ подписи и открытые параметры KMS (например, KPAK, определенный в [IETF RFC 6507]). См. рисунок D.3.

EAP-TLS изменяется следующим образом:

1) тот же, что и в EAP-TLS;

2) по получении ответа EAP с идентификатором UE – ID_UE;

3) AU передает ID_UE в RSF для проверки;

4) RSF проверяет ID_UE на основе сохраненного списка аннулирования;

5) RSF возвращает результат проверки AU;

6) если ID_UE действителен, то AU передает UE сообщение запуска EAP-TLS;

7–9) те же, что описаны в вышеупомянутом TLS-IBS;

10) EAP-Success.

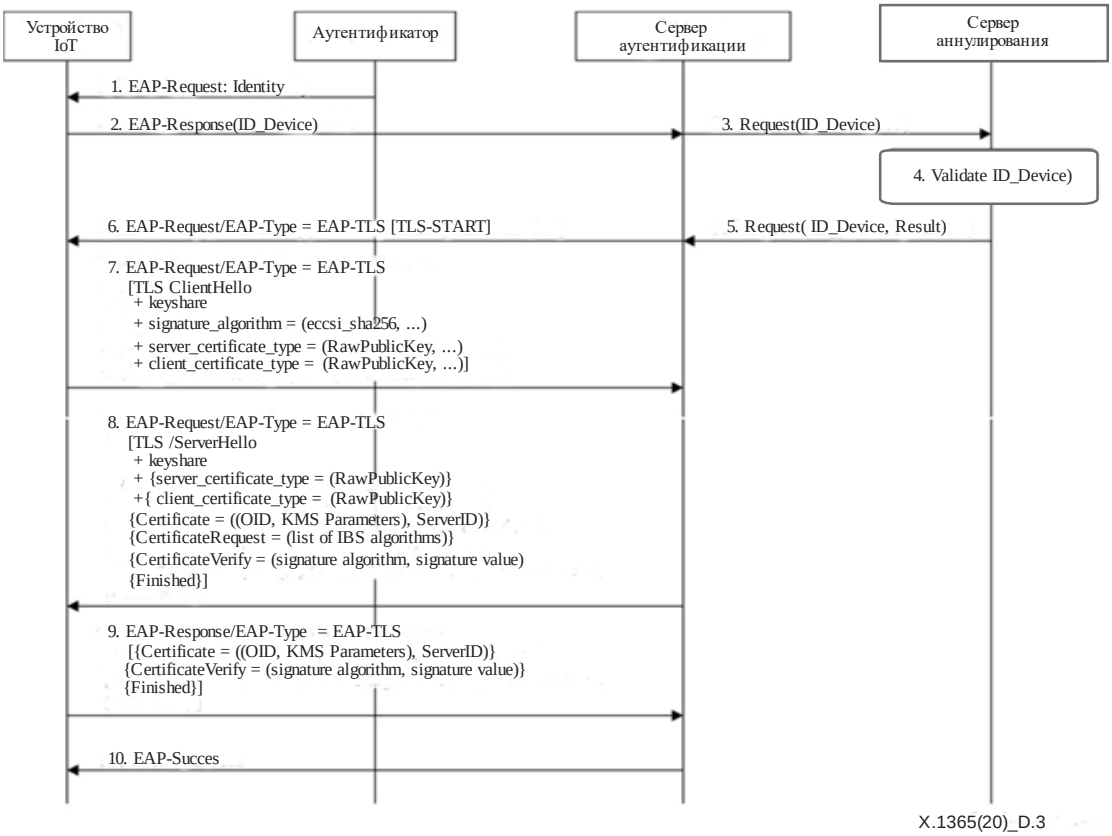

Рисунок D.3 – EAP-TLS-IBS

D.3.1 Сообщение EAP-Request

Сообщение EAP-Request имеет такой же формат, какой указан в [IETF RFC 5216].

D.3.2 Сообщение EAP-Response

Сообщение EAP-Response имеет такой же формат, какой указан в [IETF RFC 5216].

D.3.3 Сообщение ClientHello

Сообщение ClientHello имеет тот же формат, что и в пункте D.2.1.

D.3.4 Сообщение ServerHello

Сообщение ServerHello имеет тот же формат, что и в пункте D.2.2.

D.3.5 Сертификат сервера

Сертификат сервера имеет тот же формат, что и в пункте D.2.3.

D.3.6 Сертификат клиента

Сертификат клиента имеет тот же формат, что и в пункте D.2.4.

D.4 EAP-PSK-ECCSI

В данном пункте приводится описание EAP-PSK, расширенного для поддержки одного из алгоритмов аутентификации IBS – ECCSI. UE и AU имеют удостоверения на основе идентичности, которые включают идентификационные данные, SSK, PVT, KPAK, как определено в [IETF RFC 6507], в качестве параметра вычисления.

С помощью этих удостоверений UE и AU могут выработать симметричные ключи на основе статического протокола Диффи–Хеллмана путем обмена идентификационной информацией и PVT, а затем использовать SSK, принадлежащий каждому объекту. Например, UE может выработать ключ после получения идентификационных данных AU и его PVT, обозначаемых соответственно ID_AU и PVT_AU, следующим образом:

K_UE = [SSK_UE](KPAK+[hash(G || KPAK || ID_AU || PVT_AU)]PVT_AU)),

где G – точка генерирования на эллиптической кривой, используемая KMS для генерирования ключей для UE и сетей. UE и AU получают ее от KMS вместе с SSK, PVT, KPAK и т. д. Использование хеш-функции может соответствовать Приложению A [IETF RFC 6507].

Аналогично AU также может получить K_AU после получения идентификационных данных и PVT от UE следующим образом:

K_AU = [SSK_AU](KPAK+[hash(G || KPAK || ID_UE || PVT_UE)]PVT_UE)).

Можно доказать, что K_UE фактически равен K_AU.

При указанных выше свойствах EAP-PSK можно использовать для взаимной аутентификации следующим образом:

1) UE направляет AU запрос Attach Request, указывая, что для взаимной аутентификации должен использоваться EAP-PSK;

2) AU проверяет тип аутентификации и определяет метод аутентификации;

3) AU передает UE первое сообщение о EAP-PSK с полем идентичности, содержащим ID_AU и PVT_AU, а также случайное число RAND_S, как того требует EAP-PSK;

4) UE вычисляет симметричный ключ как K = [SSK_UE](KPAK+[hash(G || KPAK || ID_AU || PVT_AU)]PVT_AU)). UE генерирует случайное число RAND_P и далее получает K = KDF(K, RAND_P, RAND_S). UE вычисляет ключ аутентификации (AK) и ключ выработки ключей (KDK) на основе [b-IETF RFC 4764] для EAP-PSK;

5) UE передает AU второе сообщение о EAP-PSK, содержащее RAND_S, RAND_P, MAC_P (MAC_P = CMAC-AES-128(AK, ID_P || ID_S || RAND_S || RAND_P)), для аутентификации и поле идентичности, состоящее из ID_UE и PVT_UE;

6) AU передает ID_UE в RSF для проверки;

7) RSF проверяет ID_UE в соответствии со своим списком аннулирования;

8) RSF возвращает результат проверки AU;

9) если идентификатор действителен, то AU вычисляет симметричный ключ как K = [SSK_AU](KPAK + [hash(G || KPAK || ID_UE || PVT_UE)]PVT_UE)). Затем AU вычисляет K = KDF(K, RAND_P, RAND_S). AU вычисляет AK и KDK на основе [IETF RFC 4764] для EAP-PSK. AU аутентифицирует UE на основе MAC_P, полученного из сообщения. Затем AU вычисляет сеансовый ключ на основе RAND_P и KDK;

10) AU передает UE третье сообщение о EAP-PSK, содержащее MAC_S (MAC_S = CMAC-AES-128(AK, ID_S || RAND_P)) для аутентификации и другие поля, требуемые для EAP-PSK;

11) UE аутентифицирует AU с помощью полученного MAC_S и вычисляет сеансовый ключ с помощью RAND_P и KDK, вычисленных ранее;

12) UE передает AU последнее сообщение о EAP-PSK, завершая процедуру аутентификации EAP-PSK.

См. рисунок D.4.

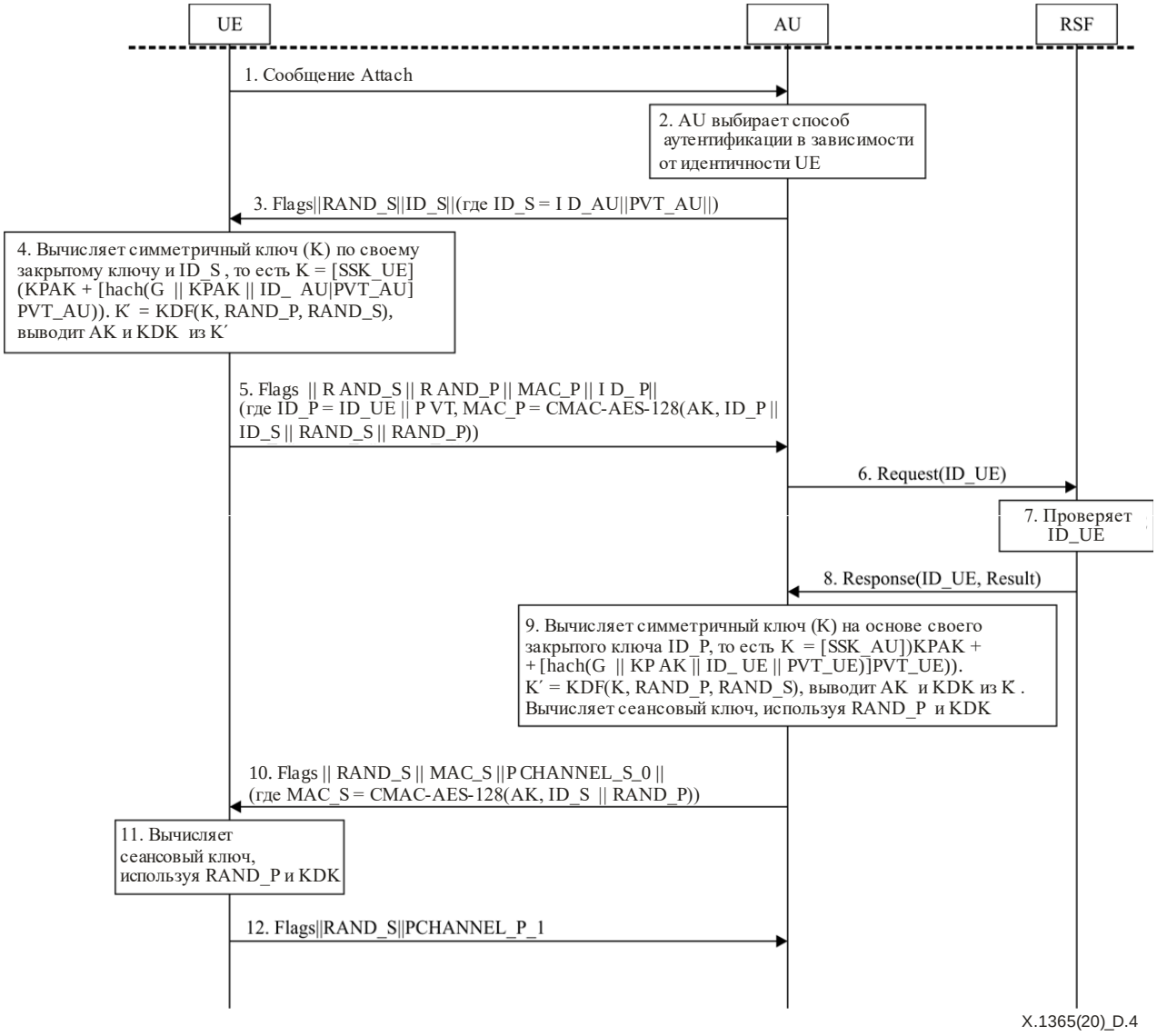

Рисунок D.4 – EAP-PSK – ECCSI

D.4.1 Сообщение Attach

Это сообщение имитирует процедуру аутентификации.

D.4.2 Первое сообщение EAP-PSK – ECCSI (сообщение 3 на рисунке D.4)

Первое сообщение EAP-PSK – ECCSI направляется сервером узлу-партнеру. Оно имеет следующий формат.

Первое сообщение EAP-PSK – ECCSI содержит:

1-байтовое поле флагов (Flags);

16-байтовое случайное число RAND_S;

поле переменной длины, в котором передается NAI сервера – ID_S. Длина этого поля выводится из поля длины EAP. Длина NAI не должна превышать 966 байтов. Это ограничение направлено на то, чтобы избежать проблем фрагментации.

На рисунке D.5 показан пример формата первого сообщения о EAP-PSK.

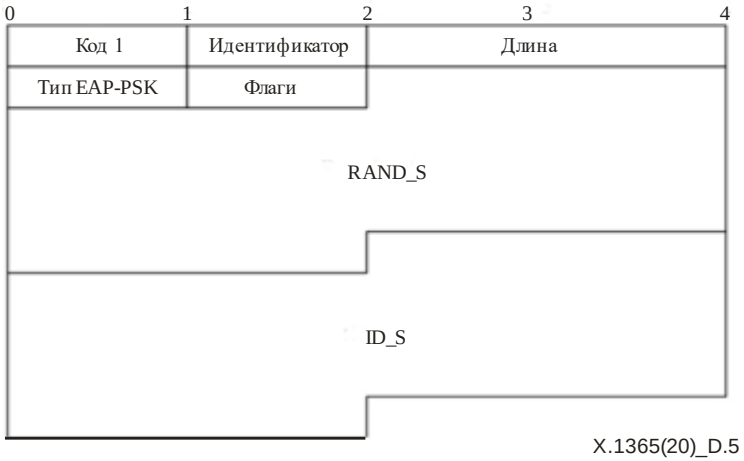

Рисунок D.5 – Формат EAP-PSK

В целях поддержки аутентификации EAP-PSK на основе IBC для переноса ID_AU и PVT_AU используется ID_S для протокола EAP-PSK. ID_S и PVT_AU переносятся в структуре данных, состоящей из тега, длины и вектора (TLV), первый октет которой содержит тег-индикатор, а второй – поле длины, указывающее длину следующего поля. В поле вектора содержится значение.

В таблице D.2 определены TLV для ID и PVT, используемые с EAP-PSK.

Таблица D.2 – Определение тега, длины и вектора для идентификатора и открытого маркера проверки

	Тег	**Длина**	**Значение**
Идентификатор	1	Переменная (≤ 255)	Определяется поставщиком услуг
PVT	2	65	Шестнадцатеричное число

На рисунке D.6 показан формат сообщения EAP-PSK – ECCSI, содержащего идентификатор и PVT в поле ID_S.

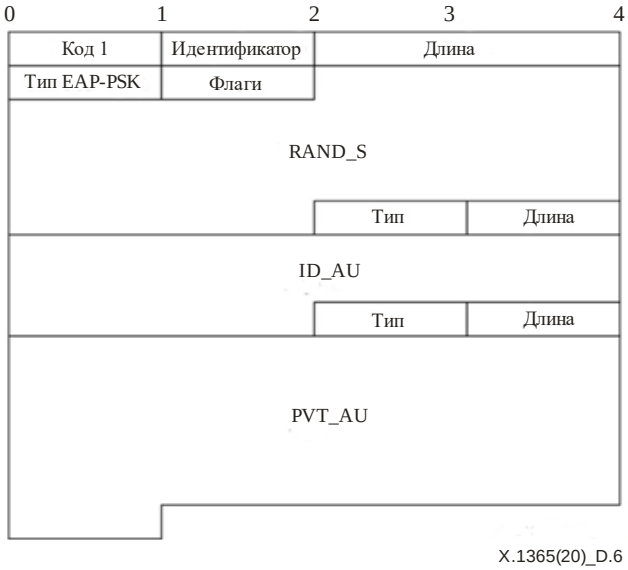

Рисунок D.6 – Формат сообщения EAP-PSK – ECCSI

D.4.3 Второе сообщение EAP-PSK – ECCSI (сообщение 5 на рисунке D.4)

Второе сообщение EAP-PSK – ECCSI передается узлом-партнером на сервер. Его формат:

– 1-байтовое поле флагов;

– 16-байтовое случайное число, переданное сервером в первом сообщении EAP-PSK – ECCSI (RAND_S), которое служит идентификатором сеанса;

– 16-байтовое случайное число RAND_P;

– 16-байтовый код управления доступом к среде передачи (MAC) – MAC_P;

– поле переменной длины, в котором передается NAI партнера – ID_P. Длина этого поля выводится из поля длины EAP. Длина NAI не должна превышать 966 байтов.

Аналогично, поле ID_S EAP-PSK используется для передачи ID_UE и PVT_UE. Формат второго сообщения EAP-PSK показан на рисунке D.7.

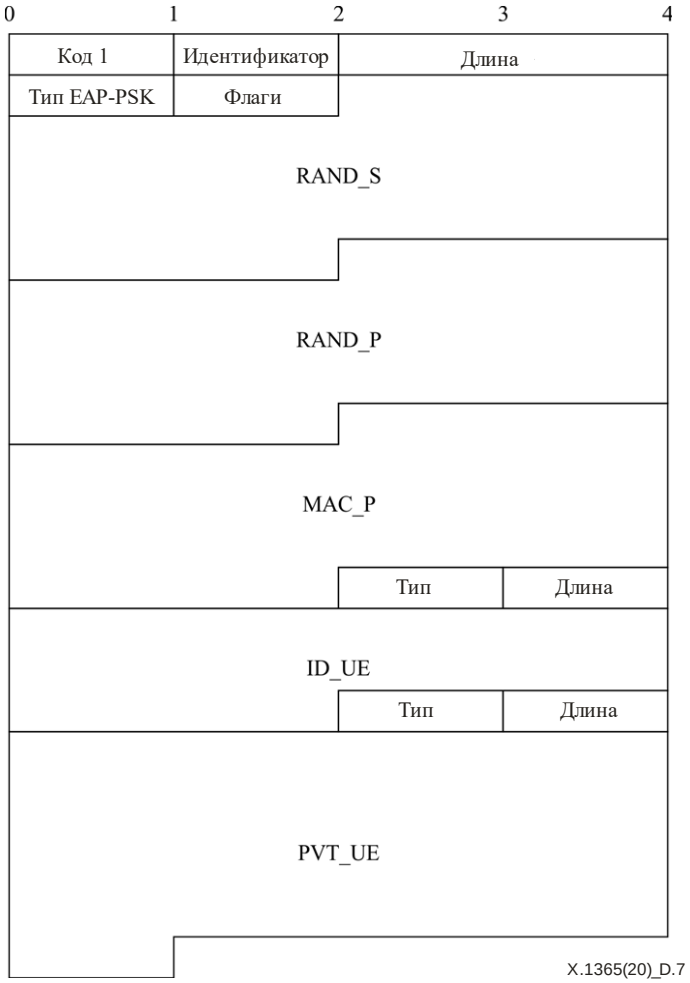

Рисунок D.7 – Формат второго сообщения EAP-PSK – ECCSI

D.4.4 Третье сообщение EAP-PSK – ECCSI (сообщение 10 на рисунке D.4)

Третье сообщение EAP-PSK – ECCSI передается сервером узлу-партнеру. Формат такой же, какой представлен в [IETF RFC 4764].

D.4.5 Четвертое сообщение EAP-PSK – ECCSI (сообщение 12 на рисунке D.4)

Четвертое сообщение EAP-PSK – ECCSI передается узлом-партнером на сервер. Формат такой же, какой представлен в [IETF RFC 4764].

Дополнение I

Именование идентичностей

(Данное Дополнение не является неотъемлемой частью настоящей Рекомендации)

Идентификатор в приложении IoT может быть идентификатором терминала или идентификатором платформы IoT. Идентификатор – это имя, которое служит для целей идентификации. Идентификатор является удобным представлением объекта и позволяет ссылаться на объект, например, в базе данных или в протоколах связи или обращаться к нему. Для достижения этой цели идентификаторы должны быть уникальными, то есть идентификатор уникален в независимой системе. Например, почтовый индекс является уникальным в той или иной стране, уникальность идентификатора задается в определенной области. Кроме того, идентификатор может присваиваться не только одному объекту, но и группе объектов, что обеспечивает единообразное управление этой группой и ее эксплуатацию.

Идентификаторы OID [b-ITU-T X.660], [b-ITU-T X-Sup.31], разработанные ИСО/МЭК совместно с МСЭ-Т, имеют множество характеристик. OID имеет иерархическую древовидную структуру, в которой можно гибко расширять уровни и длину идентификаторов. OID соответствует узлу дерева OID, который может идентифицировать что угодно (физическое или виртуальное, устройства или объекты, не являющиеся устройствами) и связывать это с глобальными информационно-коммуникационными инфраструктурами. Корень дерева содержит следующие три дуги: 0 (МСЭ-Т), 1 (ИСО) и 2 (совместно ИСО и МСЭ-Т). Каждый узел дерева представлен последовательностью целых чисел, разделенных точками, указывающей путь от корня к узлу через последовательность узлов-предков. Уровень идентификатора каждого органа регистрации должен быть назначен регистрирующим органом верхнего уровня. Например, OID, соответствующий Китайскому национальному центру регистрации IC-карт, 1.2.156.20005, выделен из OID Китайского национального центра регистрации OID 1.2.156 (ISO.member.china).

Полный OID представляет собой комбинацию идентификатора органа регистрации и идентификатора объекта, и эти две составные части разделены точкой, как показано на рисунке I-1. Если компания зарегистрировала OID в органе регистрации верхнего уровня, то ей необходимо разработать только идентификатор объекта.

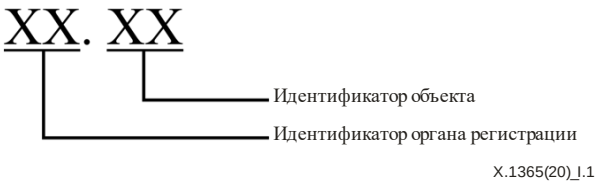

Рисунок I.1 – Структура полных OID объектов

Например, идентификатор объекта должен иметь структуру, указанную в таблице I.1.

Таблица I.1 – Подробная информация идентификатора объекта

Байт	Составная часть	Интерпретация
1	Версия и зарезервированная часть	4 бита версии идентификатора объекта и 4 бита зарезервированных цифр на будущее
2	Предприятие	Тип предприятия
3~11	Срок действия	Срок действия идентификатора, 5 байтов времени выдачи в коде времени Unix и 4 байта срока действия в секундах
12	Тип	Значение 0 – число без определенного значения, 1 – MAC и 2 – IMSI
13	Длина (величина *l*)	Размер значащей части (байты), 6 для MAC и 8 для IMSI
14~13 + *l*	Значение	Индивидуальный идентификационный номер

Идентификатор объекта имеет длину 19 байтов при использовании MAC в качестве индивидуального идентификационного номера и длину 21 байт при использовании IMSI. IMSI обычно представляется в виде 15-значного или более короткого номера с ненулевой первой цифрой, за исключением тестовой сети [b-ITU-T E.212]. Для IMSI достаточно 8 байтов с заполняющими нулями перед IMSI до 16 цифр и с использованием 4 битов для одной цифры.

Платформа IoT поддерживает список адресов. Когда оконечное устройство регистрируется впервые, платформа добавляет строку, содержащую как идентификатор, так и IP-адрес устройства. Путем поиска идентификатора устройства в списке можно получить IP-адрес, соответствующий устройству. См. таблицу I.2.

Таблица I.2 – Пример списка адресов

Идентификатор	IP-адрес
1.2.9c.4e25.10.1.5b3e408003c26700.1.6.38B1DBC3156F	192.168.0.1

Дополнение II

Расширения KMIP для поддержки IBC

(Данное Дополнение не является неотъемлемой частью настоящей Рекомендации)

KMIP можно расширить указанным ниже образом для поддержки требуемых операций IBC с KMS, в частности инициализации системы с помощью KMIP и генерирования закрытых ключей с использованием операций KMIP, как определено в разделах C.1 и C.4 соответственно.

Полезная нагрузка запроса для создания пары ключей составляется, как показано в таблице II.1.

Таблица II.1 – Полезная нагрузка запроса

Объект	Обязательный	Описание
Шаблон-атрибут закрытого ключа	Да	Задает атрибуты, когда функция **IBSetup** генерирует *ib.msk* и *ib.pubparam*

Шаблон-атрибут закрытого ключа содержит атрибуты, перечисленные в таблице II.2.

Таблица II.2 – Шаблон-атрибут закрытого ключа

Объект	Обязательный	Способ кодирования	Описание
Криптографический алгоритм	Да	Выбирается из списка, см. таблицу II.3	Определяет функцию **IBSetup**
Криптографическая длина	Нет	Целое	Определяет длину в битах характеристики простого поля, на котором основана эллиптическая кривая
Маска криптографического применения	Да	Целое	Определяет применение *ib.msk*, которое для генерирования ключа должно быть Sign (подпись). По сути **IBExtract** – это процесс подписания
Параметры криптографической области	Да	Объект	Определяет дополнительные параметры для выбора системных параметров, таких как используемая эллиптическая кривая
Криптографические параметры	Да	Объект	Указывает другие функции, такие как хеш-функция, которые должны использоваться с функциями **IBExtract**

Криптографический алгоритм принимает одно из значений, перечисленных в таблице II.3.

Таблица II.3 – Криптографический алгоритм (генерирование ключей)

Имя	Значение
IBC-KGA-BB1	00000030
IBC-KGA-BF	00000031
IBC-KGA-ECCSI	00000032
IBC-KGA-SK	00000033
IBC-KGA-SM9	00000034

Значение криптографической длины должно быть не меньше 110.

Маска криптографического применения устанавливается равной 00000001 (подпись).

Параметры криптографической области содержат атрибуты, перечисленные в таблице II.4.

Таблица II.4 – Параметры криптографической области

Объект	Обязательный	Способ кодирования	Описание
QLength	Нет	Целое	Определяет длину в битах порядка группы, из которой выбрана *ib.msk*
Рекомендуемая кривая	Да	Выбирается из списка, см. таблицу II.5	Определяет используемую кривую
Тип спаривания	Нет	Выбирается из списка, см. таблицу II.6	Если используется, то указывает тип спаривания в алгоритме на основе идентичности
Имя домена	Нет	TEXT STRING	Задает уникальное имя генерированных системных параметров *ib.pubparam*
Порядковый номер домена	Нет	INTEGER	Задает номер версии генерированных системных параметров *ib.pubparam*

Рекомендуемая кривая принимает одно из значений, перечисленных в таблице II.5.

Таблица II.5 – Рекомендуемая кривая

Имя	Значение
IBC-CURVE-SS1	00000070
IBC-CURVE-SS2	00000071
IBC-CURVE-BN-254-1	00000072
IBC-CURVE-BN-256-1	00000073
IBC-CURVE-BN-256-2	00000074
IBC-CURVE-BN-382-1	00000077
IBC-CURVE-BLS-12-381-1	0000007A
IBC-CURVE-BLS-12-442-1	0000007B
IBC-CURVE-BLS-12-455-1	0000007C
IBC-CURVE-BLS-12-461-1	0000007D
IBC-CURVE-KSS-16-340-1	0000007E
IBC-CURVE-KSS-18-348-1	0000007F

Тип спаривания принимает одно из значений, перечисленных в таблице II.6.

Таблица II.6 – Тип спаривания

Имя	Значение
Образование пар по Вейлю	00000001
Образование пар по Тейту	00000002
Оптимальное спаривание по Атэ	00000003

Криптографические параметры содержат атрибуты, перечисленные в таблице II.7.

Таблица II.7 – Криптографические параметры

Объект	Обязательный	Способ кодирования	Описание
Алгоритм хеширования	Да	Выбирается из списка, см. таблицу II.8	Определяет хеш-функцию, которую следует использовать с функцией генерирования ключей
Группа закрытых ключей	Нет	Выбирается из списка, см. таблицу II.9	Указывает, в какой группе генерируется закрытый ключ, если используется спаривание

Алгоритм хеширования принимает одно из значений, перечисленных в таблице II.8.

Таблица II.8 – Криптографический алгоритм (хеширование)

Имя	Значение
SHA224	00000040
SHA256	00000041
SHA384	00000042
SHA512	00000043
SHA3-224	00000044
SHA3-256	00000045
SHA3-384	00000046
SHA3-512	00000047
SM3	00000048

Группа закрытых ключей принимает одно из значений, указанных в таблице II.9.

Таблица II.9 – Группа закрытых ключей

Имя	Значение
IBC-PRK-GROUP1	00000001
IBC-PRK-GROUP2	00000002
IBC-PRK-TWOGROUPS	00000003

Полезная нагрузка ответа на запрос на создание пары ключей составляется так, как показано в таблице II.10.

Таблица II.10 – Полезная нагрузка ответа

Объект	Обязательный	Описание
Уникальный идентификатор закрытого ключа	Да	Уникальный идентификатор вновь созданного объекта закрытого ключа, который можно использовать для доступа к *ib.msk*. Идентификатор кодируется текстовой строкой
Уникальный идентификатор открытого ключа	Да	Уникальный идентификатор вновь созданного объекта открытого ключа, который можно использовать для доступа к *ib.pubparam*. Идентификатор кодируется текстовой строкой

Полезная нагрузка запроса операции get составляется так, как показано в таблице II.11.

Таблица II.11 – Полезная нагрузка запроса

Объект	Обязательный	Описание
Уникальный идентификатор открытого ключа	Да	Уникальный идентификатор объекта открытого ключа, который можно использовать для доступа к *ib.pubparam.* Идентификатор кодируется текстовой строкой

Полезная нагрузка ответа get составляется так, как показано в таблице II.12.

Таблица II.12 – Полезная нагрузка ответа

Объект	Обязательный	Описание
Тип объекта	Да	Тип объекта
Уникальный идентификатор	Да	Уникальный идентификатор объекта
Открытый ключ	Да	Структура открытого ключа, инкапсулирующая данные открытых параметров IBC *ib.pubparam*

Уникальный идентификатор должен совпадать с уникальным идентификатором открытого ключа, переданного в полезной нагрузке запроса get.

Тип объекта должен быть 00000003 (открытый ключ).

Блок ключей в поле открытого ключа составляется так, как показано в таблице II.13.

Таблица II.13 – Блок ключей в поле открытого ключа

Объект	Обязательный	Способ кодирования	Описание
Тип формата ключа	Да	Выбирается из списка, см. таблицу II.14	Определяет формат значения ключа
Сжатие ключа	Нет	Выбирается из списка	Указывает, следует ли сжимать значение ключа
Значение ключа	Да	Прозрачная структура ключей для открытых параметров IBC	Вновь определенная прозрачная структура ключа для открытого ключа IBC
Криптографический алгоритм	Да	Выбирается из списка, см. таблицу II.15	То же, что и полезная нагрузка запроса на создание пары ключей

Тип формата ключа соответствует значению из таблицы II.14.

Таблица II.14 – Тип формата ключа

Имя	Значение
Прозрачные открытые параметры IBC	00000016

Сжатие ключа принимает значение 00000001 (без сжатия) или 00000002 (сжатое простое число).

Значение ключа содержит атрибуты, перечисленные в таблице II.15.

Таблица II.15 – Значение ключа

Объект	Обязательный	Способ кодирования	Описание
P	Нет	Большое целое	Для кривых, основанных на простом поле, P – характеристика (p) простого поля
Q	Нет	Большое целое	Q – порядок подгруппы точек (*G*1), в которой вычисляются криптографические операции
J	Нет	Большое целое	J – кофактор, такой что J * Q = X – 1, где X – порядок группы точек заданной кривой
P1 STRING	Да	BYTE STRING	Для алгоритмов на основе спаривания P1 – это генератор группы *G*1 спаривания. Для алгоритма, не основанного на спаривании, P1 – генератор подгруппы рабочих точек
P2 STRING	Нет	BYTE STRING	Для алгоритмов на основе спаривания P2 – это генератор группы *G*2 спаривания
sP1 STRING	Нет	BYTE STRING	sP1 – скалярный результат [*ib.msk*]P1 или скалярный результат целочисленного компонента *ib.msk* с P1
sP2 STRING	Нет	BYTE STRING	Для алгоритмов на основе спаривания sP2 – скалярный результат [*ib.msk*]P2 или скалярный результат целочисленного компонента *ib.msk* с P2
sP3 STRING	Нет	BYTE STRING	Для некоторых алгоритмов на основе спаривания (в частности алгоритмов, использующих функцию генерирования ключей BB1) sP3 – это скалярный результат другого целочисленного компонента *ib.msk* с P1
Public Pairing STRING	Нет	BYTE STRING	Для некоторых алгоритмов на основе спаривания открытое спаривание – это результат pairing(P1, [s]P2), или pairing([s]P1, P2), или pairing(P1, P2), где s – *ib.msk* для таких алгоритмов, как SM9, SK-KEM или ([s1]P1, [s2]P2) для BB1-KEM, где s1, s2 – целочисленные компоненты *ib.msk*

Определения новых тегов даны в таблице II.16

Таблица II.16 – Определения тегов

Объект	Значение тега
Тип спаривания	420100
Группа закрытых ключей	420101
Имя домена	420102
Порядковый номер домена	420103
P1 STRING	420104
sP1 STRING	420105
P2 STRING	420106
sP2 STRING	420107
sP3 STRING	420108
Открытое спаривание STRING	420109

Полезная нагрузка запроса подписи составляется так, как показано в таблице II.17.

Таблица II.17 – Полезная нагрузка запроса подписи

Объект	Обязательный	Описание
Уникальный идентификатор	Нет	Уникальный идентификатор управляемого криптографического объекта, который представляет собой ключ *ib.msk* для использования в операции **IBExtract**. Если он отсутствует, то сервер использует в качестве уникального идентификатора значение заполнителя идентификатора
Криптографические параметры	Нет	Криптографические параметры могут указывать группу, из которой должен генерироваться закрытый ключ
Данные	Да	Данные указывают значение идентификационных данных, из которых извлекается закрытый ключ

Криптографические параметры содержат атрибуты, перечисленные в таблице II.18.

Таблица II.18 – Криптографические параметры

Объект	Обязательный	Способ кодирования	Описание
Группа закрытых ключей	Нет	Выбирается из списка, см. таблицу II.9	Указывает группу (*G*1 или *G*2), из которой должен генерироваться закрытый ключ

Библиография

[b-ITU-T E.101] Рекомендация МСЭ-Т E.101 (2009 год), *Определения терминов, используемых в Рекомендациях серии E для идентификаторов (наименований, номеров, адресов и других идентификаторов) служб и сетей электросвязи общего пользования.*

[b-ITU-T E.212] Рекомендация МСЭ-Т E.212 (2016 год), *План международной идентификации для сетей общего пользования и абонентов*.

[b-ITU-T X.509] Рекомендация МСЭ-Т X.509 (2019 год), *Информационные технологии – Взаимосвязь открытых систем – Справочник: Структуры сертификатов открытых ключей и атрибутов*.

[b-ITU-T X.660] Рекомендация МСЭ-Т X.660 (2011 год), *Информационные технологии – Процедуры для работы органов регистрации идентификаторов объектов: Общие процедуры и верхние дуги дерева международных идентификаторов объектов.*

[b-ITU-T X.1361] Рекомендация МСЭ-Т X.1361 (2018 год), *Структура безопасности интернета вещей на основе модели с использованием шлюза*.

[b-ITU-T X-Sup.31] ITU-T X-series Recommendations – Supplement 31 (2017), *ITU-T X.660 – Supplement on guidelines for using object identifiers for the Internet of things.*

[b-ITU-T Y.2720] Рекомендация МСЭ-Т Y.2720 (2009 год), *Структура управления определением идентичности в СПП.*

[b-ITU-T Y.4000] Рекомендация МСЭ-Т Y.4000/Y.2060 (2012 год), *Обзор интернета вещей.*

[b-ITU-T Y.4100] Рекомендация МСЭ-Т Y.4100/Y.2066 (2014 год), *Общие требования к интернету вещей.*

[b-ISO/IEC 9798-3] ISO/IEC 9798-3:2019. *IT Security techniques – Entity authentication – Part 3: Mechanisms using digital signature techniques.*

[b-ETSI TR 118 508] ETSI TR 118 508 V1.0.0 (2014), *Analysis of Security Solutions for the oneM2M System*.
<https://www.etsi.org/deliver/etsi_tr/118500_118599/118508/01.00.00_60/tr_118508v010000p.pdf>

[b-ETSI TS 133.501] ETSI TS 133 501 V15.2.0 (2018), *5G; Security architecture and procedures for 5G system (3GPP TS 33.501 version 15.1.0 Release 15)*.
<https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/15.01.00_60/ts_133501v150100p.pdf>

[b-GM/T 0044.2] GM/T 0044.2-2016, *Identity-based cryptographic algorithms SM9 – Part 2: Digital signature algorithm.*

[b-GSMA SGP.02] GSMA Official Document SGP.02 Version 3.1 (2016), *Remote Provisioning Architecture for Embedded UICC – Technical Specification.*

[b-IANA TLS REG] Internet Assigned Numbers Authority (IANA), *Transport Layer Security (TLS) Parameters*. Website available, last viewed 2019-07-12.
<https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml>

[b-IEEE 1363] IEEE 1363-2000, *IEEE Standard Specifications for Public-Key Cryptography.*

[b-IEEE P1363.3] IEEE P1363.3/D9 (May 2013), *IEEE Standard for Identity-Based Cryptographic Techniques using Pairings.*

[b-IETF RFC 3748] IETF RFC 3748 (2004). *Extensible Authentication Protocol (EAP).*

[b-OASIS KMIP] OASIS (2016), *Key Management Interoperability Protocol Specification Version 1.3.*
<http://docs.oasis-open.org/kmip/spec/v1.3/os/kmip-spec-v1.3-os.pdf>

[b-Barreto] Barreto, P. S. L. M., Libert, B., McCullagh, N., Quisquater, J.-J. (2005). Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In: Roy B. (ed.). *Advances in Cryptology – ASIACRYPT 2005, pp. 515-532. Lecture Notes in Computer Science*, vol. 3788. Berlin: Springer.

[b-Chen] Chen, L., Malone-Lee, J. (2005). Improved identity-based signcryption. In: Vaudenay S. (ed). *Public Key Cryptography – PKC 2005, pp. 362-379. Lecture Notes in Computer Science*, vol. 3386. Berlin: Springer.

[b-Ducas] Ducas, L., Lyubashevsky, V., Prest, T. (2014). Efficient identity-based encryption over NTRU lattices. In: Sarkar P., Iwata T. (eds). *Advances in Cryptology – ASIACRYPT 2014, pp. 22-41. Lecture Notes in Computer Science*, vol. 8874. Berlin: Springer.

[b-Freeman] Freeman, D., Scott, M., Teske, E. (2010). A taxonomy of pairing-friendly elliptic curves. *J. Cryptol.* **23**, pp. 224–280.

[b-Galbraith] Galbraith, S.D., Paterson, K.G., Smart, N.P. (2008). Pairings for cryptographers. *Discrete Appl. Math.*, **156**, pp. 3113-3121.

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 протокола Интернет, сети последующих поколений, интернет вещей и "умные" города
Серия Z	Языки и общие аспекты программного обеспечения для систем электросвязи

Отпечатано в Швейцарии
Женева, 2020 год