ГОСУДАРСТВЕННЫЙ СТАНДАРТ
РЕСПУБЛИКИ БЕЛАРУСЬ

СТБ 34.101.79-2019

Информационные технологии и безопасность
КРИПТОГРАФИЧЕСКИЕ ТОКЕНЫ

Інфармацыйныя тэхналогіі і бяспека
КРЫПТАГРАФІЧНЫЯ ТОКЕНЫ

Госстандарт
Минск

УДК МКС 35.240.40

Ключевые слова: криптографический токен, идентификационные данные, аутентификация, электронная цифровая подпись, прикладная программа

Предисловие

Цели, основные принципы, положения по государственному регулированию и управлению в области технического нормирования и стандартизации установлены Законом Республики Беларусь «О техническом нормировании и стандартизации».

1 РАЗРАБОТАН учреждением Белорусского государственного университета «Научно-исследовательский институт прикладных проблем математики и информатики»

ВНЕСЕН Оперативно-аналитическим центром при Президенте Республики Беларусь

2 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ постановлением Госстандарта Республики Беларусь от 8 июля 2019 года № 42

3 ВВЕДЕН ВПЕРВЫЕ

Издан на русском языке

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РЕСПУБЛИКИ БЕЛАРУСЬ

Информационные технологии и безопасность КРИПТОГРАФИЧЕСКИЕ ТОКЕНЫ

Інфармацыйныя тэхналогіі і бяспека КРЫПТАГРАФІЧНЫЯ ТОКЕНЫ

Information technology and security
Cryptographic tokens

Дата введения 2019-10-01

1 Область применения

Настоящий стандарт устанавливает архитектуру криптографического токена (КТ) и правила его использования в информационных системах для аутентификации владельца, предъявления идентификационных данных владельца, выработки электронной цифровой подписи, расшифрования ключей защиты данных. В стандарте определяются объекты, которые размещаются на КТ, сервисы, которые предоставляет КТ, протоколы работы с КТ.

Настоящий стандарт применяется при разработке средств криптографической защиты информации.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие технические нормативные правовые акты в области технического нормирования и стандартизации (далее — ТНПА):

СТБ 34.101.19-2012 Информационные технологии и безопасность. Форматы сертификатов и списков отозванных сертификатов инфраструктуры открытых ключей

СТБ 34.101.21-2009 Информационные технологии. Интерфейс обмена информацией с аппаратно-программным носителем криптографической информации (токеном)

СТБ 34.101.27-2011 Информационные технологии и безопасность. Требования безопасности к программным средствам криптографической защиты информации

СТБ 34.101.31-2011 Информационные технологии. Защита информации. Криптографические алгоритмы шифрования и контроля целостности

СТБ 34.101.45-2013 Информационные технологии и безопасность. Алгоритмы электронной цифровой подписи и транспорта ключа на основе эллиптических кривых

СТБ 34.101.47-2017 Информационные технологии и безопасность. Криптографические алгоритмы генерации псевдослучайных чисел

СТБ 34.101.66-2014 Информационные технологии и безопасность. Протоколы формирования общего ключа на основе эллиптических кривых

СТБ 34.101.77-2016 Информационные технологии и безопасность. Алгоритмы хэширования

СТБ 34.101.78-2019 Информационные технологии и безопасность. Профиль инфраструктуры открытых ключей

ГОСТ 34.973-91 (ИСО 8824-87) Информационная технология. Взаимосвязь открытых систем. Спецификация абстрактно-синтаксической нотации версии 1 (АСН.1)

ГОСТ 34.974-91 (ИСО 8825-87) Информационная технология. Взаимосвязь открытых систем. Описание базовых правил кодирования для абстрактно-синтаксической нотации версии 1 (АСН.1)

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ТНПА по каталогу, составленному по состоянию на 1 января текущего года, и по соответствующим информационным указателям, опубликованным в текущем году. Если ссылочные ТНПА заменены (изменены), то при пользовании настоящим стандартом следует руководствоваться действующими взамен ТНПА. Если ссылочные ТНПА отменены без замены, то положение, в котором дана ссылка на них, применяется в части, не затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применяют термины, установленные в СТБ 34.101.19, СТБ 34.101.27, СТБ 34.101.31, СТБ 34.101.45, СТБ 34.101.66, а также следующие термины с соответствующими определениями:

3.1 аутентификация: Проверка подлинности стороны.

3.2 билет: Информационный объект, который выпускается по итогам успешной аутентификации или после предъявления другого билета и подтверждает определенные события, факты или состояние.

3.3 билет аутентификации: Билет, который содержит утверждения аутентификации и, возможно, другие утверждения.

3.4 владелец: Пользователь, владеющий криптографическим токеном.

3.5 защищенное соединение: Соединение, которое обеспечивает конфиденциальность, контроль целостности и, возможно, подлинности сообщений.

Примечание — Контроль подлинности сообщений от стороны A к стороне B обеспечивается после того, как B провела аутентификацию A.

3.6 идентификационный атрибут: Компонент идентификационных данных.

3.7 идентификационные данные: Данные, которые однозначно характеризуют определенную сторону в определенном контексте.

Примечание — В разных контекстах могут использоваться различные идентификационные данные одной и той же стороны.

3.8 клиентская программа: Программа, которая организует взаимодействие между терминалом, пользователем и его криптографическим токеном.

3.9 команда (криптографического токена): Сообщение, которое посылается криптографическому токену и которое содержит описание определенного сервиса, предоставляемого токеном, и входные данные сервиса.

Примечание — Команда сопровождается ответом, который содержит выходные данные сервиса.

3.10 криптографический токен: Средство криптографической защиты информации, имеющее конкретного владельца и выступающее от его лица при взаимодействии с другими сторонами.

Примечание — В настоящем стандарте криптографический токен хранит один или несколько личных ключей владельца и реализует операции с ними. На токене могут размещаться идентификационные данные владельца.

3.11 прикладная программа (криптографического токена): Набор объектов и сервисов криптографического токена, имеющий определенное назначение.

3.12 терминал: Сторона, которая взаимодействует с криптографическим токеном по защищенному соединению после аутентификации перед ним и, возможно, встречной аутентификации токена.

3.13 утверждение: Характеристика стороны или события, в том числе данные о пользователе или данные о прохождении аутентификации.

Примечание — Данные о пользователе — это его идентификационные данные. Данные о прохождении аутентификации — это утверждения аутентификации.

4 Обозначения и сокращения

4.1 Обозначения

В настоящем стандарте применяют обозначения, установленные в ГОСТ 34.973, а также следующие обозначения:

$\perp$	специальный объект или ситуация: пустое слово, ошибка;
$\{0,1\}^n$	множество всех слов длины n в алфавите $\{0,1\}$;
$\{0,1\}^*$	множество всех слов конечной длины в алфавите $\{0,1\}$ (включая пустое слово $\perp$ длины 0);
$\lvert u\rvert$	длина слова $u \in \{0,1\}^*$;
$\{0,1\}^{n*}$	множество всех слов из $\{0,1\}^*$, длина которых кратна n;
α^n	для $\alpha \in \{0,1\}$ слово длины n из одинаковых символов α;
$\langle u\rangle_n$	для $u \in \{0,1\}^*$ слово из первых n символов u, $n \leqslant \lvert u\rvert$;
$u \parallel v$	конкатенация $u_1u_2\dots u_nv_1v_2\dots v_m$ слов $u = u_1u_2\dots u_n$ и $v = v_1v_2\dots v_m$;
(символы 0–F)$_{16}$	представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, при котором последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = \mathtt{A2}_{16}$);
$x \bmod m$	для целого числа x и натурального числа m остаток от деления x на m, т. е. число $r \in \{0,1,\dots,m-1\}$ такое, что m делит $x - r$;
$\lfloor u\rceil$	а) для $u = u_1u_2\dots u_8 \in \{0,1\}^8$ число $2^7u_1 + 2^6u_2 + \dots + u_8$ и б) для $u = u_1 \parallel u_2 \parallel \dots \parallel u_n$, $u_i \in \{0,1\}^8$, число $\lfloor u_1\rceil + 2^8\lfloor u_2\rceil + \dots + 2^{8(n-1)}\lfloor u_n\rceil$;
$\langle U\rangle_{8n}$	для целого числа U слово $u \in \{0,1\}^{8n}$ такое, что $\lfloor u\rceil = U \bmod 2^{8n}$;
$u \wedge v$	для $u = u_1u_2\dots u_n \in \{0,1\}^n$ и $v = v_1v_2\dots v_n \in \{0,1\}^n$ слово $w = w_1w_2\dots w_n \in \{0,1\}^n$ из символов $w_i = u_i * v_i$ (посимвольное И);
$u \vee v$	для $u = u_1u_2\dots u_n \in \{0,1\}^n$ и $v = v_1v_2\dots v_n \in \{0,1\}^n$ слово $w = w_1w_2\dots w_n \in \{0,1\}^n$ из символов $w_i = (u_i * v_i + u_i + v_i) \bmod 2$ (посимвольное ИЛИ);
$c \leftarrow u$	присвоение переменной c значения u;
$c \xleftarrow{R} U$	случайный равновероятный (или псевдослучайный) выбор c из множества U;
$A \to B$	для сторон A и B передача сообщения от A к B;

$[\text{текст}]$	необязательное сообщение (действие) протокола;
$\{\text{текст}\}$	обязательное сообщение (действие) протокола, которое может быть передано (выполнено) предварительно, до сеанса протокола, или неявно;
$\texttt{Cert}(Id, Q)$	сертификат, связывающий идентификационные данные Id определенной стороны с ее открытым ключом Q;
`hello`	приветственное сообщение;
$\langle\!\langle d_1, d_2, \ldots, d_n \rangle\!\rangle$	кодовое представление структурированных данных, содержащих компоненты $d_1, d_2, \ldots, d_n$.

4.2 Пояснения к обозначениям

4.2.1 Слова

Двоичные слова представляют собой последовательности символов из алфавита $\{0, 1\}$. Символы нумеруются слева направо от единицы. В настоящем подразделе в качестве примера рассматривается слово

$$w = 10110001100101001011101011001000.$$

В этом слове первый символ — 1, второй — 0, ..., последний — 0.

Слова разбиваются на тетрады из четверок последовательных двоичных символов. Тетрады кодируются шестнадцатеричными символами по следующим правилам (см. таблицу 1):

Таблица 1

тетрада	символ	тетрада	символ	тетрада	символ	тетрада	символ
0000	$\texttt{0}_{16}$	0001	$\texttt{1}_{16}$	0010	$\texttt{2}_{16}$	0011	$\texttt{3}_{16}$
0100	$\texttt{4}_{16}$	0101	$\texttt{5}_{16}$	0110	$\texttt{6}_{16}$	0111	$\texttt{7}_{16}$
1000	$\texttt{8}_{16}$	1001	$\texttt{9}_{16}$	1010	$\texttt{A}_{16}$	1011	$\texttt{B}_{16}$
1100	$\texttt{C}_{16}$	1101	$\texttt{D}_{16}$	1110	$\texttt{E}_{16}$	1111	$\texttt{F}_{16}$

Пары последовательных тетрад образуют октеты. Последовательные октеты слова w имеют вид:

$$10110001 = \texttt{B1}_{16},\ 10010100 = \texttt{94}_{16},\ 10111010 = \texttt{BA}_{16},\ 11001000 = \texttt{C8}_{16}.$$

4.2.2 Слова как числа

Октету $u = u_1 u_2 \ldots u_8$ ставится в соответствие байт — число $\lfloor u \rceil = 2^7 u_1 + 2^6 u_2 + \ldots + u_8$. Например, октетам w соответствуют байты

$$177 = 2^7 + 2^5 + 2^4 + 1,\ 148 = 2^7 + 2^4 + 2^2,\ 186 = 2^7 + 2^5 + 2^4 + 2^3 + 2^1,\ 200 = 2^7 + 2^6 + 2^3.$$

Число ставится в соответствие не только октетам, но и любому другому двоичному слову, длина которого кратна 8. При этом используется распространенное для многих современных процессоров соглашение «от младших к старшим» (little-endian): считается, что первый байт — младший, последний — старший. Например, слову w соответствует число

$$\lfloor w \rceil = 177 + 2^8 \cdot 148 + 2^{16} \cdot 186 + 2^{24} \cdot 200 = 3367670961.$$

4.2.3 Обозначения протоколов

Протоколы настоящего стандарта выполняют две стороны. Объекты и атрибуты (ключи, сообщения, идентификаторы, переменные) определенной стороны снабжаются нижним индексом, указывающим на эту сторону.

Квадратными скобками окаймляются необязательные сообщения и действия сторон протокола.

Фигурными скобками окаймляются сертификаты (см. раздел 9) и приветственные сообщения, которые могут передаваться предварительно или неявно. Действия по передаче и обработке таких сообщений также окаймляются фигурными скобками.

4.2.4 Обозначения данных

В настоящем стандарте в некоторых случаях применяется двухступенчатая схема описания данных. На первой ступени данные описываются рамочно, без исчерпывающих деталей. Запись $D = \langle\!\langle d_1, d_2, \ldots, d_n \rangle\!\rangle$ означает, что слово $D \in \{0,1\}^*$ является кодовым представлением структурированных данных, содержащих, по крайней мере, компоненты $d_1, d_2, \ldots, d_n$ (например, $\mathtt{Cert}(Id, Q) = \langle\!\langle Id, Q \rangle\!\rangle$). На второй ступени уточняются список компонентов, правила кодирования компонентов и всей структуры в целом.

В настоящем стандарте уточнения даются с помощью АСН.1. В других ТНПА уточнения могут быть даны другими способами. Уточнения с помощью АСН.1 собраны в модуль, представленный в приложении А.

4.3 Сокращения

В настоящем стандарте применяют следующие сокращения:
АСН.1 – абстрактно-синтаксическая нотация версии 1 (ГОСТ 34.973);
КП – клиентская программа;
КТ – криптографический токен;
ПС – прикладная система (СТБ 34.101.78);
СИ – служба идентификации (СТБ 34.101.78);
УЦ – удостоверяющий центр (СТБ 34.101.19);
ЭЦП – электронная цифровая подпись.

5 Общие положения

5.1 Назначение

КТ представляет владельца при его взаимодействии с ПС. На токене хранятся идентификационные данные владельца, ключи, сертификаты, другие объекты (см. разделы 6, 7, 9). Токен реализует криптографические алгоритмы и протоколы (см. раздел 8), с помощью которых владелец защищает доступ к своим объектам, доказывает свою подлинность перед другими сторонами, проверяет подлинность других сторон, генерирует личные ключи, подписывает данные, снимает защиту с ключей шифрования данных, выполняет другие криптографические операции. Связанные операции и объекты КТ оформляются в виде прикладных программ, которые сами являются объектами КТ. Обязательными являются программы eID и eSign (см. 6.7, 6.8).

Важно, что операции с личным ключом (выработка ЭЦП, снятие защиты) выполняются в пределах защищенной криптографической границы КТ после предъявления корректного пароля. Личный ключ не покидает пределов границы, при аппаратной защите границы секретность ключа сохраняется даже при эксплуатации КТ в агрессивной среде.

КТ используется в двух режимах — базовом и терминальном. В базовом режиме владелец настраивает КТ и выполняет с его помощью локальные (не требующие онлайн-взаимодействия с другими сторонами) криптографические операции. Взаимодействие с КТ поддерживает КП. В терминальном режиме КТ взаимодействует с ПС онлайн. Представителями систем при этом являются терминалы, взаимодействие со стороны владельца снова поддерживает КП.

В терминальном режиме КТ обеспечивает решение следующих задач:

1) аутентификация владельца на доступ к ресурсам и сервисам токена;

2) управление объектами, размещенными в пределах криптографической границы токена;

3) аутентификация терминала и аутентификация перед ним;

4) установка защищенного соединения с терминалом;

5) проверка полномочий терминала и передача ему идентификационных данных владельца;

6) проверка полномочий терминала и открытие ему доступа к прикладным программам КТ.

Получив доступ к прикладной программе КТ, терминал может выполнять примерно те же криптографические операции, что и владелец в базовом режиме. Однако в базовом режиме, несмотря на защиту личного ключа, сохраняется угроза подмены обрабатываемых (например, подписываемых) на нем данных. Для сравнения: в терминальном режиме КТ получает данные от терминала, предварительно прошедшего аутентификацию перед токеном. Данные передаются по защищенному соединению, и их невозможно раскрыть или подменить даже при полном контроле среды эксплуатации токена. Таким образом, терминальный режим обеспечивает большие гарантии безопасности при условии, что среда эксплуатации терминала защищена.

Терминал может представлять не непосредственно ПС, с которой взаимодействует владелец КТ, а специализированную СИ. Эта служба оказывает одной или нескольким ПС услугу аутентификации с одновременным предоставлением идентификационных данных. СИ по запросу ПС проводит аутентификацию на уровне «терминал СИ — КТ» и, в случае успеха, выпускает билет аутентификации. В этом билете приводятся утверждения о владельце и об аутентификации. Схема выпуска билета аутентификации определена в разделе 10.

5.2 Исполнение

КТ может быть выполнен в виде смарт-карты, USB-токена или программы, которая работает на защищенном сервере. Программные токены могут быть полезны для организации криптографических сервисов информационных систем.

Настоящий стандарт определяет командный интерфейс взаимодействия с КТ, который унифицирует работу с токенами различных типов. Перечень допустимых команд интерфейса определяется текущим состоянием КТ. Состояния описаны в разделе 11, командный интерфейс — в разделе 12.

Программное обеспечение КТ должно соответствовать требованиям СТБ 34.101.27 или аналогичных ТНПА. На аппаратном КТ должны быть предусмотрены средства инженерной защиты от злоумышленника, завладевшего токеном. Эти средства должны препятствовать чтению критических данных (личных ключей, паролей, идентификационных данных) и переводу КТ в необнаружимое небезопасное состояние.

5.3 Терминалы

Терминалы бывают двух типов — локальные и удаленные.

Локальный терминал взаимодействует с КТ непосредственно. КП является его частью. Локальный терминал должен быть аппаратным.

Удаленный терминал взаимодействует с КТ через коммуникационные сети, как правило сети Интернет. Посредником при взаимодействии, кроме КП, может быть еще и браузер.

Для организации взаимодействия с КТ терминалы снабжаются сертификатами (см. раздел 9).

5.4 Клиентская программа

КП, организуя взаимодействие между КТ, владельцем и терминалом, обрабатывает критические данные (пароли), а также открытые данные, целостность и подлинность которых определяют надежность взаимодействия.

КП не может самостоятельно обеспечить полную защиту обрабатываемых данных. Защита организуется также средствами среды эксплуатации, в том числе через настройки операционной системы. Основные задачи защиты средствами среды: невозможность чтения критических областей памяти вредоносными программами; невозможность подмены открытых данных, защита канала «браузер — КП». Организация защиты выходит за рамки настоящего стандарта.

При работе с аппаратным КТ к уязвимым критическим данным относятся только пароли доступа к КТ. Поэтому аппаратным КТ следует отдавать предпочтение в тех ситуациях, когда КП выполняется в потенциально агрессивной среде, например на компьютере, подключенном к сети общего пользования.

6 Объекты

6.1 Генератор случайных или псевдослучайных чисел

В состав КТ должен входить физический генератор случайных чисел. Генератор должен удовлетворять требованиям СТБ 34.101.27 или другого профильного ТНПА. Генератор должен использоваться для создания личных и секретных ключей, может использоваться для создания синхропосылок.

Вместо генератора случайных чисел может применяться его аналог — алгоритм генерации псевдослучайных чисел, определенный в СТБ 34.101.47 или в другом ТНПА. Входные данные алгоритма генерации должны включать долговременный секретный ключ и уникальную синхропосылку. Уровень стойкости алгоритмов, в которых планируется использовать генерируемые ключи, не должен превышать битовую длину ключа алгоритма генерации.

6.2 Таймер

Для определения текущей даты на КТ должен устанавливаться аппаратный таймер, либо если таймер поддержать нельзя, то текущая дата должна приближенно оцениваться по реквизитам входящих сертификатов или по другим данным, передаваемым на КТ от терминала. Например, текущая дата всегда позже даты выпуска очередного сертификата терминала, признанного КТ действительным.

При отсутствии аппаратного таймера оценка текущей даты должна устанавливаться при выпуске КТ в обращение равной дате выпуска.

6.3 Пароли

КТ должен поддерживать три пароля — PIN, CAN, PUK. Пароли используются в протоколе BPACE (см. 8.3) и являются общими для всех прикладных программ КТ. Пароли записываются на КТ при выпуске токена в обращение. Пароли PIN и PUK конфиденциально передаются владельцу, CAN передается в открытом виде.

Пароль PIN (от Personal Identification Number) представляет собой случайное число из 6 десятичных цифр, известное только владельцу КТ. Используется для контроля доступа к данным и прикладным программам КТ. Может быть изменен владельцем в процессе эксплуатации КТ после ввода верного действующего PIN.

PIN снабжается счетчиком попыток, который первоначально равен 3. При неверном вводе PIN счетчик уменьшается на 1. Если счетчик достигает значения 1, то PIN приостанавливается и далее требуется ввести CAN. Ввод CAN не изменяет счетчик. При верном CAN пароль PIN возобновляется — его снова можно ввести. При неверном CAN доступ к КТ временно блокируется. Если счетчик попыток достигает значения 0, то блокируется PIN. При вводе верного PIN счетчик попыток возвращается к значению 3.

PIN может быть разблокирован после ввода верного PUK. Однако если при заблокированном PIN пароль PUK вводится неверно 10 раз, то PIN блокируется навсегда. При успешной разблокировке PIN счетчик попыток возвращается к значению 3.

PIN может быть деактивирован и повторно активирован с помощью специальных команд. Для деактивации требуется предъявить верный PIN или PUK, для активации — верный PUK. Сразу после выпуска токена PIN активирован. При деактивации PIN доступ к операциям и данным, требующим аутентификации по PIN, запрещен. В том числе запрещена аутентификация по PIN, разблокировка PIN. Деактивация и повторная активация PIN не изменяют ни статус его блокировки, ни счетчик попыток.

Пароль CAN (от Card Access Number) представляет собой число из 6 десятичных цифр, которое не может быть вычислено на основании общей информации о КТ (например, серийном номере) или его владельце. Может быть напечатан на корпусе КТ или указан в сопроводительных документах.

Пароль CAN не может быть заблокирован или изменен. Он используется для защиты от атак типа «отказ в обслуживании». Защита состоит в требовании ввести CAN перед последней проверкой PIN. Дополнительно CAN может использоваться для получения доступа к функциям и данным прикладной программы eID авторизованным терминалом, т. е. терминалом, который был успешно аутентифицирован с помощью протокола BAUTH (см. 8.4) и в сертификате которого установлено соответствующее право (см. 7.3).

Пароль PUK (от PIN Unlock Key) представляет собой случайное число из 10 десятичных цифр, известное только владельцу КТ. PUK не может быть заблокирован или изменен. Используется для разблокировки PIN. Дополнительно может использоваться для деактивации и активации PIN.

После ввода неверного CAN или PUK следует временно заблокировать КТ не менее чем на 1 с. Для контроля времени блокировки следует использовать аппаратный таймер либо организовать вычисления требуемой продолжительности.

При передаче в команды КТ пароли PIN, CAN и PUK представляются строками октетов, в которых октет 30_{16} кодирует цифру 0 пароля, октет 31_{16} — цифру 1, . . . , октет 39_{16} — цифру 9.

6.4 Личные ключи

На КТ обязательно хранится личный ключ КТ, который используется для аутентификации КТ перед терминалом. Ключ устанавливается при выпуске КТ в обращение вместе с сертификатом соответствующего открытого ключа. Процедуры выпуска КТ должны гарантировать, что ключ сохраняется исключительно в пределах криптографической границы токена. К личному ключу КТ можно обратиться только косвенно, через выполнение протокола аутентификации. Ключ не может быть прочитан с КТ или изменен.

Дополнительно на КТ могут храниться личные ключи, которые используются в прикладной программе eSign для выработки подписи и (или) разбора токена ключа. Личные ключи eSign генерируются самим КТ при выпуске в обращение или эксплуатации. Личный ключ не может быть прочитан с КТ, но может быть уничтожен или сгенерирован заново. Ключ, сгенерированный в определенном режиме (базовом или терминальном), может использоваться только в этом режиме. Личный ключ генерируется вместе с открытым.

Предусмотрены 6 личных ключей eSign, которым назначены идентификаторы $\mathtt{01}_{16}$, $\mathtt{02}_{16}$, $\mathtt{03}_{16}$, $\mathtt{11}_{16}$, $\mathtt{12}_{16}$, $\mathtt{13}_{16}$. Идентификатор должен использоваться для выбора определенного личного ключа перед выполнением криптографической операции с ним. Первые 3 идентификатора соответствуют базовому режиму, последние 3 — терминальному. Второй символ идентификатора кодирует уровень стойкости алгоритмов СТБ 34.101.45: $\ell = 128$ кодируется единицей, $\ell = 192$ — двойкой и $\ell = 256$ — тройкой.

Определенный в СТБ 34.101.21 программный интерфейс Cryptoki должен быть следующим образом настроен на работу с личными ключами eSign:

1 Должны действовать соглашения СТБ 34.101.78.

2 Атрибут `CKA_ID` должен быть одним из 6 указанных выше идентификаторов.

3 При генерации ключей (механизм `CKM_EC_KEY_PAIR_GEN`) уровень стойкости, определяемый атрибутом `CKA_ID`, должен соответствовать параметрам эллиптической кривой, указываемым в атрибуте `CKA_EC_PARAMS` открытого ключа.

4 Атрибут `CKA_SENSITIVE` должен принимать значение `CK_TRUE`, и это значение не может изменяться.

5 Атрибут `CKA_EXTRACTABLE` должен принимать значение `CK_FALSE`, и это значение не может изменяться.

6 Атрибуты `CKA_SIGN`, `CKA_UNWRAP` личного ключа и атрибуты `CKA_VERIFY`, `CKA_WRAP` открытого ключа должны принимать значение `CK_TRUE`.

7 Атрибут `CKA_ALLOWED_MECHANISMS` должен включать механизм `CKM_BIGN_TSP`, а также механизм `CKM_BIGN_HBELT` при уровне стойкости $\ell = 128$ или механизм `CKM_BIGN_BASH` при уровне стойкости $\ell = 192$ и $\ell = 256$.

6.5 Сертификаты

Открытый ключ Q стороны с идентификационными данными Id распространяется в форме сертификата $\mathtt{Cert}(Id, Q)$. Сертификат представляет собой объект вида $\langle\!\langle Id, Q \rangle\!\rangle$, который связывает идентификационные данные с открытым ключом и, косвенно, личным ключом стороны. Связывание выполняет УЦ, подписывая данные сертификата на своем личном ключе. Соответствующий открытый ключ УЦ также распространяется в форме сертификата, который выпускается либо вышестоящим УЦ, либо является корневым (самоподписанным). В целом образуется маршрут сертификации — цепочка, которая начинается корневым сертификатом и заканчивается $\mathtt{Cert}(Id, Q)$.

При выпуске КТ в обращение на него записывается сертификат КТ, представляющий личный ключ токена. На КТ записывается также один или несколько корневых сертификатов, которые будут использоваться при аутентификации терминала.

В процессе аутентификации терминал предъявляет свой маршрут сертификации без первого (корневого) сертификата. КТ проверяет маршрут, в том числе проверяет, что недостающий корневой сертификат был записан на КТ. В ответ КТ отправляет терминалу свой сертификат. Терминал самостоятельно восстанавливает маршрут сертификации КТ и после этого проверяет его.

Чтобы уменьшить объем пересылаемых данных и упростить проверку сертификатов, они делаются облегченными относительно стандартных сертификатов, определенных в СТБ 34.101.19. Формат облегченных сертификатов определен в разделе 9. Там же определены правила проверки маршрутов сертификации.

В настоящем стандарте требуется, чтобы длина маршрута сертификации равнялась 2 или 3. Другими словами, облегченные сертификаты КТ и терминалов должны выпускаться либо непосредственно корневым УЦ, либо промежуточным УЦ, подчиненным корневому.

УЦ, выпускающие облегченные сертификаты, должны снабжаться уникальными номерами из трех десятичных цифр. УЦ с определенным номером должен использовать только одну пару ключей (личный и открытый).

Дополнительно на КТ могут записываться сертификаты, представляющие личные ключи eSign. Сертификаты eSign являются стандартными, их содержание и формат определены в СТБ 34.101.19. Сертификаты должны соответствовать дополнительным требованиям, установленным в СТБ 34.101.78 для роли «Физические лица». В приложении Б даны рекомендации по хранению сертификатов eSign в пределах криптографической границы КТ.

6.6 Объект `Name`

При формировании сертификата eSign готовится запрос, в который включаются специальным образом отформатированные идентификационные данные. Формат определяется типом `Name`, заданным в СТБ 34.101.19. Должны применяться уточнения формата, установленные в СТБ 34.101.78.

Объект `Name` готовится терминалом по прочитанным идентификационным данным, или объект записывается на КТ в процессе выпуска токена в обращение или в процессе эксплуатации. В приложении Б даны рекомендации по хранению объекта в пределах криптографической границы КТ.

6.7 Прикладная программа eID

Прикладная программа eID предназначена для управления идентификационными атрибутами владельца КТ со стороны терминала, представляющего некоторую информационную систему. Управление атрибутами осуществляется в соответствии с правами, установленными владельцем КТ, и правами, заданными в маршруте сертификации терминала (см. 9.2). Атрибуты передаются по защищенному соединению и таким образом обеспечивается их конфиденциальность, контролируется целостность и подлинность.

Программа eID может использоваться только в терминальном режиме после успешной аутентификации терминала.

Программе назначается идентификатор `id-eID`, определенный в приложении А. В приложении Б даны рекомендации по хранению объектов eID в пределах криптографической границы КТ.

6.8 Прикладная программа eSign

Прикладная программа eSign организует работу с криптографическими алгоритмами СТБ 34.101.45. Программа обеспечивает генерацию личных и открытых ключей, управление соответствующими сертификатами, выработку подписи и разбор токена ключа. С помощью последовательностей команд eSign организуются сложные макрооперации, например, процесс выпуска сертификатов открытых ключей алгоритмов ЭЦП/транспорта ключа.

Программа eSign может использоваться в базовом режиме после успешной аутентификации владельца либо в терминальном режиме после успешной аутентификации терминала (требуется предварительная аутентификация владельца).

Для выполнения некоторых команд eSign требуется согласие владельца КТ. Владелец выражает согласие, подтверждая знание PIN. Программа eSign должна поддерживать флаг подтверждения PIN. Флаг устанавливается после успешной аутентификации владельца по паролю PIN, а также при последующем предъявлении PIN по защищенному соединению. Флаг сбрасывается после генерации и уничтожения ключей, выработки определенного числа ЭЦП, изменения PIN и других операций.

Программе eSign назначается идентификатор `id-eSign`, определенный в приложении А. В приложении Б даны рекомендации по хранению объектов eSign в пределах криптографической границы КТ.

7 Идентификационные данные

7.1 Группы данных

Информация о КТ и идентификационные атрибуты владельца КТ организованы в виде групп данных DG1, DG2,. . . , DG22. Наименование, содержание и обязательность групп определяются в таблице 2. Непосредственно идентификационные атрибуты владельца КТ располагаются в группах DG4–DG12, DG17, DG18, DG21, DG22. Группы DG13, DG19, DG20 зарезервированы для будущего применения, в настоящем стандарте они не определяются.

Группы данных описываются следующими одноименными типами АСН.1. В типах применяется неявное (`IMPLICIT`) тегирование.

```
SerialNumber ::= [APPLICATION 1] PrintableString(SIZE(14..18))
IssuingState ::= [APPLICATION 2] Country
DateOfExpiry ::= [APPLICATION 3] Date
GivenName ::= [APPLICATION 4] UTF8String
FamilyName ::= [APPLICATION 5] UTF8String
MiddleName ::= [APPLICATION 6] UTF8String
PersonalNumber ::= [APPLICATION 7] PrintableString(SIZE(7..64))
DateOfBirth ::= [APPLICATION 8] Date
PlaceOfBirth ::= [APPLICATION 9] GeneralPlace
Nationality ::= [APPLICATION 10] Country
Sex ::= [APPLICATION 11] ICAOSex
OptionalDataR ::= [APPLICATION 12] SET OF OptionalData
```

Таблица 2 — Группы данных

Группа	Наименование	Содержание	Обязательность
DG1	SerialNumber	Серийный номер КТ	+
DG2	IssuingState	Страна, выпустившая КТ	+
DG3	DateOfExpiry	Дата окончания действия КТ	+
DG4	GivenName	Имя	+
DG5	FamilyName	Фамилия	+
DG6	MiddleName	Отчество	+
DG7	PersonalNumber	Идентификационный номер	+
DG8	DateOfBirth	Дата рождения	+
DG9	PlaceOfBirth	Место рождения	+
DG10	Nationality	Гражданство	+
DG11	Sex	Пол	+
DG12	OptionalDataR	Дополнительные данные для чтения	
DG13	—	Зарезервировано на будущее	
DG14	WrittenSignature	Изображение ручной подписи	
DG15	DateOfIssuance	Дата выпуска КТ	+
DG16	IssuanceBoard	Орган, выдавший КТ	+
DG17	PlaceOfResidence	Адрес постоянного места жительства	
DG18	DistrictID	Код региона	
DG19	—	Зарезервировано на будущее	
DG20	—	Зарезервировано на будущее	
DG21	PhoneNumber	Номер телефона	
DG22	EMailAddress	Адрес электронной почты	

```
WrittenSignature ::= [APPLICATION 14] OCTET STRING
DateOfIssuance ::= [APPLICATION 15] Date
IssuanceBoard ::= [APPLICATION 16] UTF8String
PlaceOfResidence ::= [APPLICATION 17] GeneralPlace
DistrictID ::= [APPLICATION 18] PrintableString(SIZE(0..64))
PhoneNumber ::= [APPLICATION 21] PrintableString
EmailAddress ::= [APPLICATION 22] IA5String

ICAOString ::= PrintableString(FROM("A".. "Z" | " "))
Country ::= ICAOString(SIZE(3))
ICAOSex ::= PrintableString(FROM("M"|"F"|" "))
Date ::= NumericString(SIZE(8))

Place ::= SEQUENCE {
  street  [10] UTF8String OPTIONAL,
  city    [11] UTF8String,
  state   [12] UTF8String OPTIONAL,
  country [13] Country,
  zipcode [14] PrintableString OPTIONAL
}
```

```
GeneralPlace ::= CHOICE {
  structuredPlace Place
  freetextPlace   [1] UTF8String
  noPlaceInfo     [2] UTF8String
}

OptionalData ::= SEQUENCE {
  type OBJECT IDENTIFIER,
  data ANY DEFINED BY type OPTIONAL
}
```

Серийный номер КТ в группе DG1 должен устанавливаться в соответствии с правилами, заданными в СТБ 34.101.78 для идентификационного атрибута `serialNumber`. Согласно этим правилам, серийный номер должен сопровождаться префиксом `'IDCBY-'` (например, `'IDCBY-590082394654'`). Длина серийного номера (без учета префикса) должна быть в диапазоне от 8 до 12 символов.

Код страны в группах DG2 и DG10 должен иметь трехсимвольное обозначение согласно [1]. В группе DG2 код страны должен иметь значение `'BLR'`. В группе DG10 должно задаваться значение, соответствующее гражданству владельца КТ (`'BLR'` для граждан Республики Беларусь), и значение `'XXX'`, если владелец КТ не имеет гражданства или его гражданство не определено.

Дата окончания действия КТ в группе DG3, дата рождения владельца КТ в группе DG8 и дата выпуска КТ в группе DG15 должны иметь формат YYYYMMDD (4 символа года, 2 символа месяца, 2 символа числа месяца). Если дата рождения известна не полностью, то пропущенные данные помечаются пробелом. Когда необходимо проверить возраст владельца КТ, вместо недостающих данных используется наиболее поздняя возможная дата (например, 31 декабря указанного года, если известен только год рождения). Год рождения в DG8 содержится всегда, даже если он в действительности точно не известен (в последнем случае указывается приблизительный год рождения).

Идентификационный номер владельца КТ в группе DG7 должен устанавливаться в соответствии с правилами, заданными в СТБ 34.101.78 для идентификационного атрибута `serialNumber`. Идентификационный номер должен начинаться с префикса `'PNOXX'`, где символы `'XX'` определяют двухсимвольный код страны согласно [1], в которой зарегистрирован номер (например, `'PNOBY-786545091A4PB5'`).

Место рождения владельца КТ в группе DG9 и адрес постоянного места жительства владельца КТ в группе DG17 рекомендуется задавать типом `Place`, компоненты которого имеют следующее значение:

1) компонент `street` должен содержать название улицы (переулка, проспекта, площади и т. п.) и уточняющие данные (номер дома, корпуса, квартиры и т. п.);

2) компонент `city` должен содержать название населенного пункта (города, поселка, деревни и т. п.);

3) компонент `state` должен содержать название области (штата, провинция и т. п.) и, при необходимости, района (округа, департамента и т. п.);

4) компонент `country` должен содержать код страны (см. описание группы DG2);

5) компонент `zipcode` должен содержать почтовый индекс владельца КТ.

Для задания адреса в произвольном виде следует воспользоваться компонентом `freetextPlace` типа `GeneralPlace`. Если владелец КТ не проживает в Республике Беларусь, то в `GeneralPlace` должен быть выбран компонент `noPlaceInfo` со значением `'He`

пражывае ў РБ/Не проживает в РБ'. Если информация о месте рождения или месте жительства владельца КТ не известна, то в GeneralPlace должен быть выбран компонент noPlaceInfo со значением 'Інфармацыя адсутнічае/Информация отсутствует'.

В группе DG14 задается изображение ручной подписи владельца КТ в кодировке JPEG 2000 [2].

В группе DG16 указывается организация, которая выдает КТ владельцу.

Код региона в группе DG18 задается строкой, которая содержит (слева направо):

– три символа типа кода региона;
– два символа кода страны согласно [1], в которой зарегистрирован код региона;
– символ '-';
– собственно символы кода региона.

Строка в DG18 должна быть пустой, если в DG17 не задан адрес постоянного места жительства.

В качестве кода региона рекомендуется использовать почтовый индекс. Этому коду назначается тип 'ZIP' (например, 'ZIPBY-220045').

Номер телефона владельца КТ в группе DG21 должен быть представлен в синтаксисе global-number согласно [3].

В группе DG22 может быть задан адрес электронной почты владельца КТ.

В группах данных DG4, DG5, DG6 информация представляется на белорусском языке, русском языке и латинице, а в группах данных DG9, DG16, DG17 — на белорусском и русском языках. Для разделения информации, представленной на разных языках, должен использоваться символ '/'. Например, в DG5 фамилия представляется в виде 'МІЦКЕВІЧ/МИЦКЕВИЧ/MITSKEVICH'. Если у владельца КТ отсутствует отчество, то группа DG6 должна быть пустой.

При хранении групп на КТ они должны кодироваться с помощью отличительных правил, описанных в СТБ 34.101.19 (приложение Б).

Доступ на запись к группам DG1–DG16 должен быть запрещен.

В приложении Б приводятся идентификаторы файлов, в которых группы данных хранятся на КТ.

7.2 Дополнительные атрибуты

При аутентификации терминал запрашивает идентификационные атрибуты владельца КТ. Кроме атрибутов, определенных в 7.1, перечень может содержать дополнительные атрибуты DocumentValidity, AgeVerification и PlaceVerification. Дополнительным атрибутам назначаются идентификаторы, заданные в приложении А.

Атрибут DocumentValidity используется для проверки срока действия КТ. Параметром атрибута является дата, на которую необходимо провести проверку. Проверка состоит в сравнении даты в группе DG3 с датой, указанной в параметре. Результат проверки будет отрицательным, если дата в параметре более поздняя. Значением атрибута является результат проверки.

Проверка срока действия КТ должна выполняться всегда и предшествовать запросу любых других атрибутов.

Атрибут AgeVerification используется для проверки возраста владельца КТ. Параметром атрибута является самая поздняя допустимая дата рождения. Проверка состоит в сравнении даты в группе DG8 с датой, указанной в параметре. Значением атрибута является признак того, что владелец КТ родился не позднее указанной в параметре даты.

Атрибут PlaceVerification используется для проверки проживания владельца КТ в определенном регионе. Параметром атрибута является код региона. Проверка состоит в сравнении кода региона в группе DG18 со значением, указанным в параметре. Значением атрибута является результат проверки.

Параметры дополнительных атрибутов должны передаваться КТ в формате, заданном следующим типом АСН.1:

```
AuthAuxData ::= [APPLICATION 7] SEQUENCE OF DiscretionaryDataTemplate

DiscretionaryDataTemplate ::= [APPLICATION 19] SEQUENCE {
  objIdentifier  OBJECT IDENTIFIER,
  dataObjects    ANY DEFINED BY objIdentifier
}
```

Компонент `objIdentifier` задает идентификатор дополнительного атрибута, а компонент `dataObjects` — параметр атрибута. Компонент `dataObjects` должен иметь уникальный контекстно-зависимый тег [см. ГОСТ 34.974 (таблица 1)]. В таблице 3 указаны допустимые теги и типы компонента `dataObjects` для определенных выше дополнительных атрибутов.

Таблица 3 — Параметры дополнительных атрибутов

Атрибут	Идентификатор	Тег	Тип параметра (АСН.1)
DocumentValidity	`id-DocumentValidity`	83_{16}	`Date`
AgeVerification	`id-AgeVerification`	88_{16}	`Date`
PlaceVerification	`id-PlaceVerification`	92_{16}	`PrintableString(SIZE(7..64))`

7.3 Права доступа

Права доступа к прикладной программе КТ описываются следующим типом АСН.1:

```
CertHAT ::= [APPLICATION 76] SEQUENCE {
  objId              OBJECT IDENTIFIER,
  discretionaryData  OCTET STRING
}
```

Компонент `objId` описывает прикладную программу. Компонент `discretionaryData` определяет слово прав доступа. Битам этого слова соответствуют определенные группы данных, дополнительные атрибуты и другие объекты, которыми управляет прикладная программа.

Для описания прав доступа к eID компонент `objId` должен принимать значение `id-eIdAccess`, определенное в приложении А. Компонент `discretionaryData` должен состоять из пяти октетов.

Эти октеты определяют двоичное слово длины 40. Биты слова нумеруются справа налево, начиная с 0. Слово определяет права доступа к идентификационным атрибутам после аутентификации по PIN. Слово также определяет право управления самим паролем PIN и право доступа к атрибутам по паролю CAN вместо PIN. Предполагается, что право проверять срок действия КТ имеет любая сторона. Поэтому дополнительный атрибут DocumentValidity в слове прав доступа не учитывается.

В таблице 4 описываются биты слова прав доступа к прикладной программе eID. Биты, зарезервированные на будущее, должны быть нулевыми.

Например, слово 00000000 00110011 01101111 01111011 00010000 = $\texttt{003C6F5B10}_{16}$ определяет права на чтение по паролям PIN и CAN групп данных DG1, DG2, DG4–DG7, DG9–DG12, DG14, DG15, DG17, DG18, DG21, DG22.

Таблица 4 — Слово прав доступа к прикладной программе eID

Номера битов	Права доступа
0	Право проверять возраст (атрибут `AgeVerification`)
1	Право проверять принадлежность к региону (атрибут `PlaceVerification`)
2, 3	Зарезервировано на будущее
4	Право получить доступ по паролю CAN
5	Право управлять паролем PIN (активировать, деактивировать, изменять, разблокировать)
6, 7	Зарезервировано на будущее
8	Право читать DG1
...	...
29	Право читать DG22
30, 31	Зарезервировано на будущее
32	Право записывать в DG22
...	...
37	Право записывать в DG17
38, 39	Зарезервировано на будущее

Для описания прав доступа к прикладной программе eSign компонент `objId` должен принимать значение `id-eSignAccess`, определенное в приложении А. Компонент `discretionaryData` должен состоять из двух октетов.

Эти октеты определяют двоичное слово длины 16. Биты слова нумеруются справа налево, начиная с 0. Слово определяет права доступа к данным и сервисам eSign после аутентификации по PIN. Слово также определяет права управления самим паролем PIN.

Примечание — Формально PIN не является объектом прикладных программ eID и eSign. Но права управления PIN удобно ввести именно через эти программы. Программа eSign, в отличие от eID, может использоваться и в базовом, и в терминальном режимах. Поэтому именно eSign делегированы основные права.

В таблице 5 описываются биты слова прав доступа к прикладной программе eSign. Биты, зарезервированные на будущее, должны быть нулевыми.

При парольной аутентификации КП передает КТ согласованные с владельцем объекты `CertHAT`, контролируя дальнейшие возможности работы с токеном. Передаваемые объекты никем не заверены, и поэтому КТ сужает заданные в них права до управления паролями и работы с eSign в базовом режиме.

В терминальном режиме КТ получает объекты `CertHAT` в сертификате терминала, заверенном цепочкой УЦ. Сертификаты УЦ также содержат объекты `CertHAT`. Слова прав КП, терминала и всех УЦ цепочки накладываются друг на друга с помощью операции $\wedge$ (логическое И). КТ переключается на права, полученные в результате наложения.

Таблица 5 — Слово прав доступа к прикладной программе eSign

Номера битов	Права доступа
0 – 5	Зарезервировано на будущее
6	Право активировать и деактивировать пароль PIN
7	Право изменять пароль PIN
8	Право разблокировать пароль PIN
9	Право записывать объект `Name`
10	Право активировать и деактивировать ключи
11	Зарезервировано на будущее
12	Право генерировать и уничтожать ключи, записывать сертификаты, читать объект `Name` в терминальном режиме
13	Право генерировать и уничтожать ключи, записывать сертификаты, читать объект `Name` в базовом режиме
14	Право вырабатывать подпись, разбирать токен ключа, читать сертификаты в терминальном режиме
15	Право вырабатывать подпись, разбирать токен ключа, читать сертификаты в базовом режиме

8 Криптографическая поддержка

8.1 Эллиптические кривые

Для выработки и проверки ЭЦП, транспорта ключа и аутентификации используются алгоритмы и протоколы на основе эллиптических кривых, описанных в СТБ 34.101.45. Во всех алгоритмах и протоколах используются параметры эллиптической кривой p, a, b, q и G, заданные в СТБ 34.101.45 (приложение Б). Этим параметрам в СТБ 34.101.45 (приложение Д) назначены идентификаторы `bign-curve256v1` (уровень стойкости $\ell = 128$), `bign-curve384v1` ($\ell = 192$), `bign-curve512v1` ($\ell = 256$).

Параметры p, a и b определяют группу точек эллиптической кривой $E_{a,b}(\mathbb{F}_p)$. Эта группа имеет порядок q, ее элементами являются аффинные точки, образующие множество $E^*_{a,b}(\mathbb{F}_p)$, и бесконечно удаленная точка. Параметр G (базовая точка) является элементом $E^*_{a,b}(\mathbb{F}_p)$. Группа записывает аддитивно. Для натурального числа u и аффинной точки V через uV обозначается сумма u экземпляров V.

При передаче между сторонами точка V представляется словом $\langle V \rangle_{4\ell} \in \{0,1\}^{4\ell}$. Правила представления заданы в СТБ 34.101.45 (пункт 4.1).

В алгоритмах и протоколах на основе эллиптических кривых используются личные и открытые ключи, долговременные или одноразовые. Личным ключом является число из множества $\{1, 2, \ldots, q-1\}$, открытым ключом — элемент множества $E^*_{a,b}(\mathbb{F}_p)$.

8.2 Стандартные криптографические алгоритмы

Алгоритм `belt-cfb`. Используется алгоритм зашифрования в режиме гаммирования с обратной связью, определенный в СТБ 34.101.31. Входными данными алгоритма являются сообщение $X \in \{0,1\}^*$, ключ $\in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Алгоритм $\texttt{belt-cfb}^{-1}$. Используется алгоритм расшифрования в режиме гаммирования с обратной связью, определенный в СТБ 34.101.31. Входными данными алгоритма

являются зашифрованное сообщение $Y \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является первоначальное сообщение $X \in \{0,1\}^{|Y|}$.

Алгоритм `belt-mac`. Используется алгоритм выработки имитовставки `belt-mac`, определенный в СТБ 34.101.31. Входными данными алгоритма являются сообщение $X \in \{0,1\}^*$ и ключ $K \in \{0,1\}^{256}$, выходными — имитовставка $T \in \{0,1\}^{64}$.

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, определенный в СТБ 34.101.31. Входными данными алгоритма является слово $X \in \{0,1\}^*$, выходными — его хэш-значение $Y \in \{0,1\}^{256}$.

Алгоритмы `bash384`, `bash512`. Используются алгоритмы хэширования `bash384`, `bash512`, определенные в СТБ 34.101.77. Входными данными алгоритмов является слово $X \in \{0,1\}^*$, выходными — его хэш-значение $Y \in \{0,1\}^{384}$ или $Y \in \{0,1\}^{512}$. КТ может не реализовывать алгоритмы, если не предполагается проверка облегченных сертификатов (см. раздел 9), подписанных на ключах уровней стойкости $\ell = 192$ и $\ell = 256$.

Алгоритм `belt-keywrap`. Используется алгоритм установки защиты ключа `belt-keywrap`, определенный в СТБ 34.101.31. Входными данными алгоритма являются защищаемый ключ $X \in \{0,1\}^{8*}$, его заголовок $I \in \{0,1\}^{128}$, ключ защиты $K \in \{0,1\}^{256}$. Выходными данными является защищенный ключ $Y \in \{0,1\}^{|X|+128}$.

Алгоритм `belt-keywrap`$^{-1}$. Используется алгоритм снятия защиты ключа `belt-keywrap`$^{-1}$, определенный в СТБ 34.101.31. Входными данными алгоритма являются защищенный ключ $Y \in \{0,1\}^{8*}$, заголовок $I \in \{0,1\}^{128}$, ключ защиты $K \in \{0,1\}^{256}$. Выходными данными является либо признак ошибки $\perp$, либо первоначальный ключ $X \in \{0,1\}^{|Y|-128}$.

Алгоритм `belt-keyrep`. Используется алгоритм преобразования ключа `belt-keyrep`, определенный в СТБ 34.101.31. Входными данными алгоритма являются преобразуемый ключ $X \in \{0,1\}^{256}$, уровень $D \in \{0,1\}^{96}$, заголовок $I \in \{0,1\}^{128}$, длина $m \in \{128, 192, 256\}$. Выходными данными является преобразованный ключ $Y \in \{0,1\}^m$.

Алгоритм `bign-genkeypair`. Используется алгоритм генерации пары ключей `bign-genkeypair`, определенный в СТБ 34.101.45. Неявными входными данными алгоритма являются параметры эллиптической кривой. В алгоритме используются случайные или псевдослучайные числа. Выходными данными алгоритма являются личный ключ $d \in \{1, 2, \ldots, q-1\}$ и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$.

Алгоритм `bign-valpubkey`. Используется алгоритм проверки открытого ключа `bign-valpubkey`, определенный в СТБ 34.101.45 (пункт 6.2.3). Входными данными алгоритма является открытый ключ $Q = (x_Q, y_Q)$, где x_Q, y_Q — целые числа. Выходными данными алгоритма является ответ `ДА` (Q является допустимым открытым ключом) или `НЕТ`.

Алгоритм `bign-sign`. Используется алгоритм выработки ЭЦП `bign-sign`, определенный в СТБ 34.101.45 с одним из алгоритмов хэширования `belt-hash` (на уровне стойкости $\ell = 128$), `bash384` ($\ell = 192$) или `bash512` ($\ell = 256$). Входными данными `bign-sign` являются сообщение $X \in \{0,1\}^*$ и личный ключ $d \in \{1, 2, \ldots, q-1\}$. Выходными данными является подпись $S \in \{0,1\}^{3\ell}$.

Алгоритм `bign-verify`. Используется алгоритм проверки ЭЦП `bign-verify`, определенный в СТБ 34.101.45 с одним из алгоритмов хэширования `belt-hash` (на уровне стойкости $\ell = 128$), `bash384` ($\ell = 192$) или `bash512` ($\ell = 256$). Входными данными `bign-verify` являются сообщение $X \in \{0,1\}^*$, подпись $S \in \{0,1\}^*$ и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$. Выходными данными является ответ `ДА` или `НЕТ`.

Алгоритм `bign-keytransport`. Используется алгоритм создания токена ключа `bign-keytransport`, определенный в СТБ 34.101.45. Входными данными алгоритма являются ключ $X \in \{0,1\}^{8*}$, его заголовок $I \in \{0,1\}^{128}$ и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$ получателя X. Длина X должна быть не меньше 128. Выходными данными является слово $Y \in \{0,1\}^{2\ell+|X|+128}$ — токен ключа X.

Алгоритм $\texttt{bign-keytransport}^{-1}$. Используется алгоритм разбора токена ключа $\texttt{bign-keytransport}^{-1}$, определенный в СТБ 34.101.45. Входными данными алгоритма являются токен $Y \in \{0,1\}^{8*}$, заголовок $I \in \{0,1\}^{128}$ транспортируемого в нем ключа и личный ключ $d \in \{1,2,\ldots,q-1\}$ получателя Y. Выходными данными является либо признак ошибки $\perp$, либо слово $X \in \{0,1\}^{|Y|-2\ell-128}$ — ключ, который транспортируется в токене Y.

Алгоритм `bake-kdf`. Используется алгоритм построения ключа `bake-kdf`, определенный в СТБ 34.101.66 (пункт 6.1.3). Входными данными алгоритма являются секретное слово $X \in \{0,1\}^*$, дополнительное слово $S \in \{0,1\}^*$ и номер ключа C — неотрицательное целое число. Выходными данными является ключ $Y \in \{0,1\}^{256}$.

8.3 Протокол BPACE

8.3.1 Общее описание

Для парольной аутентификации используется протокол формирования общего ключа BPACE, определенный в СТБ 34.101.66 (пункт 7.6). Протокол выполняется между КТ и КП, при этом КТ выступает в роли стороны A (в обозначениях СТБ 34.101.66), а КП — в роли стороны B.

Приветственное сообщение $\texttt{hello}_{\text{КТ}}$ ($\texttt{hello}_A$ в обозначениях СТБ 34.101.66) не используется, т. е. является пустым словом. Соответственно не отправляется и не обрабатывается сообщение M0. Приветственное сообщение $\texttt{hello}_{\text{КП}}$ отправляется перед выполнением протокола и поэтому сообщение M1 содержит только зашифрованные данные $Y_{\text{КП}}$ (Y_A в обозначениях СТБ 34.101.66). Даже не включенное явно в M1 сообщение $\texttt{hello}_{\text{КП}}$ должно использоваться при построении ключа.

При выполнении протокола КП и КТ должны подтверждать общий ключ, т. е. выполнять действия и передавать сообщения, помеченные в СТБ 34.101.66 квадратными скобками. Подтверждая ключ, стороны проводят аутентификацию друг друга.

8.3.2 Входные и выходные данные

Входными данными протокола является приветственное сообщение $\texttt{hello}_{\text{КП}} \in \{0,1\}^*$ и пароль $P \in \{0,1\}^{8*}$. В качестве P могут выступать пароли PIN, CAN, PUK (см. 6.3).

Выходными данными протокола является либо общий ключ $K_0 \in \{0,1\}^{256}$, либо признак ошибки $\perp$. Возврат $\perp$ означает либо сбой при передаче сообщений протокола, либо нарушение целостности сообщений, либо нарушение их подлинности, либо ошибку аутентификации стороны протокола. Общий ключ K_0 стороны используют для организации защищенного соединения.

8.4 Протокол BAUTH

8.4.1 Общее описание

Протокол BAUTH используется для взаимной аутентификации КТ и терминала. В описании протокола терминал обозначается буквой T. В ходе протокола стороны проводят аутентификацию друг друга и дополнительно формируют общий секретный ключ.

Сначала КТ проверяет подлинность терминала, а затем терминал проверяет подлинность КТ. Аутентификация КТ перед терминалом может не выполняться, соответствующие сообщения и действия сторон окаймляются квадратными скобками.

Протокол BAUTH должен выполняться после парольной аутентификации с помощью протокола BPACE.

8.4.2 Долговременные ключи

Стороны протокола используют долговременные личные ключи d_{T}, $d_{\mathrm{KT}} \in \{1, 2, \ldots, q-1\}$ и соответствующие открытые ключи Q_{T}, $Q_{\mathrm{KT}} \in E^*_{a,b}(\mathbb{F}_p)$. Для генерации пар ключей должен использоваться алгоритм `bign-genkeypair`.

При хранении и распространении должны обеспечиваться конфиденциальность и контроль целостности личных ключей, контроль целостности открытых ключей. Личные ключи d_{T}, d_{KT} дополнительно к протоколу BAUTH могут использоваться в алгоритмах ЭЦП, определенных в СТБ 34.101.45, для проверки владения личным ключом при формировании сертификатов. Использование личных ключей в других алгоритмах и протоколах запрещено.

Открытые ключи распространяются в виде сертификатов $\mathtt{Cert}(Id_{\mathrm{T}}, Q_{\mathrm{T}})$, $\mathtt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}})$. При выполнении протокола сертификаты проверяются. Сертификат должен признаваться действительным, только если корректны его формат, срок действия, подпись УЦ, выпустившего сертификат.

Протокол построен так, что терминал получает сертификат $\mathtt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}})$, только предъявляя действительный сертификат $\mathtt{Cert}(Id_{\mathrm{T}}, Q_{\mathrm{T}})$ и владея соответствующим личным ключом d_{T}.

8.4.3 Входные и выходные данные

Входными данными протокола являются приветственные сообщения $\mathtt{hello}_{\mathrm{T}}, \mathtt{hello}_{\mathrm{KT}} \in \{0,1\}^*$, личные ключи d_{T}, d_{KT} и сертификаты $\mathtt{Cert}(Id_{\mathrm{T}}, Q_{\mathrm{T}})$, $\mathtt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}})$.

Выходными данными протокола является либо общий ключ $K_0 \in \{0,1\}^{256}$, либо признак ошибки $\perp$. Возврат $\perp$ означает либо сбой при передаче сообщений протокола, либо нарушение целостности сообщений, либо нарушение их подлинности, либо ошибку аутентификации стороны протокола. Общий ключ K_0 является дополнительным результатом аутентификации, этот ключ стороны используют для организации защищенного соединения.

8.4.4 Переменные

Одноразовые личный и открытый ключи. КТ генерирует одноразовые личный ключ $u_{\mathrm{KT}} \in \{1, 2, \ldots, q-1\}$ и соответствующий открытый ключ $V_{\mathrm{KT}} \in E^*_{a,b}(\mathbb{F}_p)$. Ключ V_{KT} пересылается терминалу. Ключ u_{KT} должен быть уничтожен после использования.

Одноразовый секретный ключ. КТ генерирует и пересылает терминалу одноразовый секретный ключ $R_{\mathrm{KT}} \in \{0,1\}^{\ell}$. Ключ должен генерироваться в соответствии с требованиями 6.1.

Защищенный одноразовый ключ. Ключ R_{KT} защищается с помощью алгоритма `belt-keywrap` и пересылается в виде слова $Y_{\mathrm{KT}} \in \{0,1\}^{\ell+128}$.

Служебный ключ K. Для защиты R_{KT} используется служебный ключ $K \in \{0,1\}^{256}$, который строится по личному и открытому ключам противоположных сторон. Ключ K должен быть уничтожен после использования.

Одноразовая подпись. Для аутентификации КТ перед терминалом используется одноразовая подпись $s_{\text{КТ}} \in \{0, 1, \ldots, q-1\}$.

Зашифрованные данные. КТ пересылает терминалу одноразовую подпись и свой сертификат в зашифрованном виде. Результатом зашифрования является слово $Z_{\text{КТ}} \in \{0,1\}^*$.

Синхропосылка. При аутентификации КТ перед терминалом используется синхропосылка $R_{\text{Т}} \in \{0,1\}^{128}$. В различных сеансах протокола терминал должен вырабатывать либо заведомо различные синхропосылки, либо вероятность совпадения синхропосылок должна быть пренебрежимо мала. Синхропосылки могут вырабатываться случайным или псевдослучайным методом, строиться по меткам времени, значениям монотонного счетчика и др.

Имитовставки. Для аутентификации терминала перед КТ используется имитовставка $T_{\text{Т}} \in \{0,1\}^{64}$. Для контроля целостности зашифрованных данных используется имитовставка $T_{\text{КТ}} \in \{0,1\}^{64}$.

Служебный ключ K_1. Для вычисления имитовставок используется служебный ключ $K_1 \in \{0,1\}^{256}$. Ключ должен быть уничтожен после использования.

Служебный ключ K_2. Для шифрования одноразовой подписи и сертификата используется служебный ключ $K_2 \in \{0,1\}^{256}$. Ключ должен быть уничтожен после использования.

Переменная t. Используется переменная $t \in \{0,1\}^{\ell}$.

8.4.5 Сообщения

Стороны пересылают друг другу следующие сообщения:
M0 (Т $\to$ КТ): $\{\texttt{hello}_{\text{Т}} \parallel \}\ \texttt{Cert}(Id_{\text{Т}}, Q_{\text{Т}})$;
M1 (КТ $\to$ Т): $\{\texttt{hello}_{\text{КТ}} \parallel \}\ \langle V_{\text{КТ}} \rangle_{4\ell} \parallel Y_{\text{КТ}}$;
M2 (Т $\to$ КТ): $T_{\text{Т}}\ [\ \parallel R_{\text{Т}}]$;
M3 (КТ $\to$ Т): $[Z_{\text{КТ}} \parallel T_{\text{КТ}}]$.

8.4.6 Шаги

Аутентификация состоит в выполнении шагов, определенных ниже. При ошибке на любом из шагов, в том числе при отрицательном результате любой проверки, протокол прекращается и возвращается признак ошибки $\perp$.

1 Т:
  1) {отправляет сообщение M0}.

2 КТ:
  1) {получает сообщение M0};
  2) {проверяет $\texttt{Cert}(Id_{\text{Т}}, Q_{\text{Т}})$};
  3) $R_{\text{КТ}} \xleftarrow{R} \{0,1\}^{\ell}$ (в соответствии с требованиями 8.4.4);
  4) $(u_{\text{КТ}}, V_{\text{КТ}}) \leftarrow \texttt{bign-genkeypair}()$;
  5) $K \leftarrow \langle u_{\text{КТ}} Q_{\text{Т}} \rangle_{256}$;
  6) $Y_{\text{КТ}} \leftarrow \texttt{belt-keywrap}(R_{\text{КТ}}, 0^{128}, K)$;
  7) отправляет сообщение M1.

3 Т:
  1) получает сообщение M1;
  2) проверяет, что $\texttt{bign-valpubkey}(V_{\text{КТ}}) = \texttt{ДА}$;
  3) $K \leftarrow \langle d_{\text{Т}} V_{\text{КТ}} \rangle_{256}$;
  4) $R_{\text{КТ}} \leftarrow \texttt{belt-keywrap}^{-1}(Y_{\text{КТ}}, 0^{128}, K)$;

5) $\big[R_{\mathrm{T}} \xleftarrow{R} \{0,1\}^{128}$ (в соответствии с требованиями 8.4.4)$\big]$;
6) $K_0 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, \big[R_{\mathrm{T}} \parallel \big]\ \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 0)$;
7) $K_1 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, \big[R_{\mathrm{T}} \parallel \big]\ \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 1)$;
8) $\big[K_2 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, R_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 2)\big]$;
9) $T_{\mathrm{T}} \leftarrow \texttt{belt-mac}(0^{128}, K_1)$;
10) отправляет сообщение M2.

4 KT:
1) получает сообщение M2;
2) $K_0 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, \big[R_{\mathrm{T}} \parallel \big]\ \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 0)$;
3) $K_1 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, \big[R_{\mathrm{T}} \parallel \big]\ \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 1)$;
4) проверяет, что $T_{\mathrm{T}} = \texttt{belt-mac}(0^{128}, K_1)$;
5) $\big[K_2 \leftarrow \texttt{bake-kdf}(R_{\mathrm{KT}}, R_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{T}} \parallel \texttt{hello}_{\mathrm{KT}}, 2)\big]$;
6) $\big[t \leftarrow \langle \texttt{belt-hash}(\langle V_{\mathrm{KT}} \rangle_{2\ell} \parallel R_{\mathrm{T}}) \rangle_{\ell}\big]$;
7) $\big[s_{\mathrm{KT}} \leftarrow (u_{\mathrm{KT}} - (2^{\ell} + \lfloor t \rceil) d_{\mathrm{KT}}) \bmod q\big]$;
8) $\big[Z_{\mathrm{KT}} \leftarrow \texttt{belt-cfb}(\langle s_{\mathrm{KT}} \rangle_{2\ell} \parallel \texttt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}}), K_2, 0^{128})\big]$;
9) $\big[T_{\mathrm{KT}} \leftarrow \texttt{belt-mac}(Z_{\mathrm{KT}}, K_1)\big]$;
10) $\big[$отправляет сообщение M3$\big]$.

$\big[$5 T:
1) получает сообщение M3;
2) проверяет, что $T_{\mathrm{KT}} = \texttt{belt-mac}(Z_{\mathrm{KT}}, K_1)$;
3) $(\langle s_{\mathrm{KT}} \rangle_{2\ell} \parallel \texttt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}})) \leftarrow \texttt{belt-cfb}^{-1}(Z_{\mathrm{KT}}, K_2, 0^{128})$;
4) проверяет, что $s_{\mathrm{KT}} \in \{0, 1, \ldots, q-1\}$;
5) проверяет $\texttt{Cert}(Id_{\mathrm{KT}}, Q_{\mathrm{KT}})$;
6) $t \leftarrow \langle \texttt{belt-hash}(\langle V_{\mathrm{KT}} \rangle_{2\ell} \parallel R_{\mathrm{T}}) \rangle_{\ell}$;
7) проверяет, что $s_{\mathrm{KT}} G + (2^{\ell} + \lfloor t \rceil) Q_{\mathrm{KT}} = V_{\mathrm{KT}}\big]$.

Успешное выполнение шагов протокола означает, что КТ провел аутентификацию терминала, $\big[$терминал провел аутентификацию КТ,$\big]$ и дополнительно стороны выработали общий ключ K_0.

8.5 Защищенное соединение

8.5.1 Общее описание

При функционировании КТ могут создаваться два защищенных соединения: «КП — КТ» и «терминал — КТ». Защита означает обеспечение конфиденциальности передаваемых сообщений и контроль их целостности и подлинности. Ко всем отсылаемым сообщениям применяется алгоритм установки защиты, а ко всем входящим сообщениям — алгоритм снятия защиты.

Инициатором отправки сообщений всегда выступает КП или терминал. Сообщение от этих сторон к КТ является частью некоторой команды, описанной в разделе 12. Обратное сообщение от КТ является частью ответа на команду.

8.5.2 Ключи защиты

Для защиты сообщений используется ключ $K_0 \in \{0,1\}^{256}$, по которому строятся ключи $K_1, K_2 \in \{0,1\}^{256}$. Ключ K_1 используется для имитозащиты, ключ K_2 — для шифрования.

Алгоритм построения ключей шифрования и имитозащиты состоит из следующих шагов:

1 Установить $K_1 \leftarrow \texttt{belt-keyrep}(K_0, 0^{96}, \langle 1 \rangle_{128}, 256)$.

2 Установить $K_2 \leftarrow \texttt{belt-keyrep}(K_0, 0^{96}, \langle 2 \rangle_{128}, 256)$.

3 Возвратить (K_1, K_2).

Ключи K_0, K_1, K_2 должны быть уничтожены при закрытии соединения.

8.5.3 Счетчик

Стороны соединения используют переменную-счетчик C, которая принимает неотрицательные целые значения. При создании соединения счетчик полагается равным 0.

КП и терминал используют слово $\langle C + 1 \rangle_{128}$ в качестве синхропосылки при зашифровании исходящих (и предназначенных КТ) сообщений. Соответственно, КТ использует слово $\langle C + 1 \rangle_{128}$ для расшифрования входящих сообщений. Обратно, КТ использует $\langle C + 2 \rangle_{128}$ для зашифрования исходящих (и предназначенных КП или терминалу) сообщений. Соответственно, КП и терминал используют $\langle C + 2 \rangle_{128}$ для расшифрования входящих сообщений.

КП и терминал после отправки-приема сообщений и КТ после приема-отправки увеличивают значение C на 2: $C \leftarrow C + 2$.

При переключении с одного соединения на другое КТ должен сохранять ключи и счетчик текущего соединения, поскольку возможны обратные переключения.

8.5.4 Алгоритмы защиты

8.5.4.1 Входные и выходные данные, переменные

Входными данными алгоритма установки защиты являются заголовок $I \in \{0,1\}^{8*}$, критическое сообщение $X \in \{0,1\}^{8*}$, ключи K_1, $K_2 \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является кодовое представление $\langle\!\langle I, Y, T \rangle\!\rangle$ заголовка I, зашифрованного сообщения $Y \in \{0,1\}^{8*}$ и имитовставки $T \in \{0,1\}^{64}$.

Входными данными алгоритма снятия защиты является кодовое представление $\langle\!\langle I, Y, T \rangle\!\rangle$ заголовка I, зашифрованного сообщения $Y \in \{0,1\}^{8*}$ и имитовставки $T \in \{0,1\}^{64}$, а также ключи K_1, $K_2 \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является либо признак ошибки $\perp$, либо первоначальное сообщение $X \in \{0,1\}^{8*}$. Возврат $\perp$ означает нарушение целостности или подлинности переданных данных.

Синхропосылки алгоритмов установки и снятия защиты строятся по правилам, определенным в 8.5.3. Правила кодирования выходных и промежуточных данных определены в 12.4.

8.5.4.2 Алгоритм установки защиты

Алгоритм установки защиты состоит в выполнении шагов, определенных ниже. При ошибке на любом из шагов алгоритм прекращается и возвращается признак ошибки $\perp$.

1 Установить $Y \leftarrow \texttt{belt-cfb}(X, K_2, S)$.

2 Выполнить кодирование I, Y и построить их кодовое представление $\langle\!\langle I, Y \rangle\!\rangle$.

3 $T \leftarrow \texttt{belt-mac}(S \parallel \langle\!\langle I, Y \rangle\!\rangle, K_1)$.

4 Выполнить кодирование I, Y, T и построить их кодовое представление $\langle\!\langle I, Y, T \rangle\!\rangle$.

5 Возвратить $\langle\!\langle I, Y, T \rangle\!\rangle$.

8.5.4.3 Алгоритм снятия защиты

Алгоритм снятия защиты состоит в выполнении шагов, определенных ниже. При ошибке на любом из шагов, в том числе при отрицательном результате проверки, алгоритм прекращается и возвращается признак ошибки $\perp$.

1 Выполнить декодирование $\langle\!\langle I, Y, T \rangle\!\rangle$ и определить I, Y, T.

2 Выполнить кодирование I, Y и построить их кодовое представление $\langle\!\langle I, Y \rangle\!\rangle$.

3 Проверить, что $T = \texttt{belt-mac}(S \parallel \langle\!\langle I, Y \rangle\!\rangle, K_1)$.

4 $X \leftarrow \texttt{belt-cfb}^{-1}(Y, K_2, S)$.

5 Возвратить X.

9 Облегченные сертификаты

9.1 Формат

Облегченные сертификаты (card verifiable certificates в [4]) имеют меньший объем и более простой формат, чем стандартные сертификаты, определенные в СТБ 34.101.19.

Облегченный сертификат описывается следующими типами ACH.1. В типах применяется неявное (IMPLICIT) тегирование.

```
CVCertificate ::= [APPLICATION 33] SEQUENCE {
  certificateBody                    [APPLICATION 78] SEQUENCE {
    certProfileIdentifier            [APPLICATION 41] INTEGER {v1(0)},
    certAuthorityReference           [APPLICATION 2]  CharString,
    publicKey                        [APPLICATION 73] PubKey,
    certHolderReference              [APPLICATION 32] CharString,
    certHolderAuthorizationTemplate [APPLICATION 76] CertHAT OPTIONAL,
    certEffectiveDate                [APPLICATION 37] CVDate,
    certExpirationDate               [APPLICATION 36] CVDate,
    certExtensions                   [APPLICATION 5]  CVExt OPTIONAL
  },
  signature                          [APPLICATION 55] OCTET STRING
}

CharString ::= PrintableString(SIZE(8..12))

PubKey ::= SEQUENCE {
  objIdentifier         OBJECT IDENTIFIER,
  pubKeyandParameters   PublicKey
}

CVDate :: = OCTET STRING(SIZE(6))
CVExt ::= SEQUENCE OF DiscretionaryDataTemplate
```

Компоненты, вложенные в CVCertificate, имеют следующее значение.

1 Компонент certProfileIdentifier определяет версию формата.

2 Компонент certAuthorityReference идентифицирует эмитента (УЦ, выпустивший сертификат). Представляет собой строку 'BYCA0XXX' для корневых УЦ либо строку 'BYCA1XXX' для подчиненных, где 'XXX' — номер УЦ (см. 6.5).

3 Компонент publicKey описывает открытый ключ. Вложенный компонент objIdentifier должен принимать значение bign-pubkey, заданное в СТБ 34.101.45 (приложение Д). Открытый ключ записывается во вложенный компонент pubKeyandParameters типа PublicKey. Этот тип определен в СТБ 34.101.45 следующим образом:

```
PublicKey ::= BIT STRING(SIZE(512|768|1024))
```

По длине открытого ключа определяется уровень стойкости и стандартные параметры эллиптической кривой этого уровня (см. раздел 8).

4 Компонент `certHolderReference` идентифицирует субъекта (владельца) сертификата. Если субъектом является УЦ, то при формировании компонента повторяется логика `certAuthorityReference`. Если же субъектом является КТ или терминал, то в `certHolderReference` задается серийный номер сертификата. Длина серийного номера должна быть в диапазоне от 8 до 12 символов.

5 Компонент `certHolderAuthorizationTemplate` определяет права доступа субъекта сертификата к прикладной программе eID. Правила формирования компонента определены в 7.3.

Компонент должен отсутствовать в сертификате КТ. Отсутствие компонента в других сертификатах означает отсутствие прав доступа к eID.

6 Компонент `certEffectiveDate` определяет дату выпуска сертификата. Дата задается строкой формата `YYMMDD`, в которой `YY` — год текущего века, `MM` — месяц, `DD` — день. Символы строки — десятичные числа $0, 1, \ldots, 9$ — кодируются октетами $\mathtt{00}_{16}, \mathtt{01}_{16}, \ldots, \mathtt{09}_{16}$.

7 Компонент `certExpirationDate` определяет дату окончания действия сертификата.

8 Компонент `certExtensions` определяет список расширений, которые содержат дополнительную информацию о субъекте сертификата.

Настоящий стандарт допускает только одно расширение, которое описывает права доступа к прикладной программе eSign. Расширению назначается идентификатор `id-SignAuthExt`, определенный в приложении А. Этот идентификатор указывается в компоненте `objIdentifier` типа `DiscretionaryDataTemplate`, а в компоненте `dataObjects` указывается значение типа `CertHAT`. Правила формирования значения определены в 7.3.

Компонент `certExtensions` должен отсутствовать в сертификате КТ. Отсутствие компонента в других сертификатах означает отсутствие прав доступа к eSign.

9 Компонент `signature` содержит ЭЦП сертификата, выработанную УЦ. Уровень стойкости ключей УЦ не должен быть ниже уровня стойкости ключей субъекта сертификата.

9.2 Проверка маршрута сертификации

В протоколе аутентификации BAUTH терминал предъявляет КТ свой маршрут сертификации без первого (корневого) сертификата. Сертификаты маршрута должны предъявляться последовательно, начиная с сертификата, выпущенного корневым УЦ, и заканчивая сертификатом терминала. КТ определяет недостающий сертификат по идентификатору эмитента (`certAuthorityReference`) в первом сертификате представленной терминалом цепочки. Если самоподписанный сертификат с нужным идентификатором не был записан на КТ во время выпуска в обращение, то проверка маршрута заканчивается ошибкой, иначе найденный сертификат вставляется в первую позицию маршрута.

В ответ КТ может предъявить терминалу свой сертификат. Терминал восстанавливает маршрут сертификации КТ, используя предустановленные сертификаты УЦ или онлайн-доступ к сервисам получения таких сертификатов. Восстановление маршрута выполняется последовательно, начиная с сертификата КТ и заканчивая корневым сертификатом. Для определения очередного сертификата используется идентификатор эмитента, указанный в предыдущем. Проверка сертификата КТ завершается ошибкой, если маршрут восстановить не удалось.

При проверке маршрута его сертификаты последовательно просматриваются, начиная с корневого. Проверка проводится следующим образом:

1 Проверяется, что эмитент следующего сертификата является субъектом текущего.

2 Проверяется, что текущая дата попадает в срок действия каждого сертификата маршрута. Если проверка проводится на КТ без аппаратного таймера, то оценка текущей даты (см. 6.2) обновляется: она устанавливается равной самой поздней из дат выпуска сертификатов маршрута.

3 Проверяется, что расширения сертификатов удовлетворяют правилам, заданным в 9.1.

4 ЭЦП каждого следующего сертификата проверяется на открытом ключе текущего.

Если проверка проводится КТ, то дополнительно определяются права доступа терминала к прикладным программам токена. Слово прав доступа к eID (eSign) определяется в результате наложения слов, указанных в компонентах `certHolderAuthorizationTemplate` (расширениях `id-SignAuthExt`) сертификатов. Наложение выполняется с помощью операции $\wedge$ (логическое И).

Если проверка проводится терминалом, то дополнительно могут контролироваться статусы отзыва сертификатов маршрута. Проверка завершается ошибкой, если маршрут содержит отозванный сертификат. Проверку статуса отзыва нельзя выполнить на КТ, и поэтому сертификаты терминалов рекомендуется обновлять как можно чаще.

10 Выпуск билета аутентификации

10.1 Схема

В настоящем разделе определяется схема взаимодействия ПС, КП, КТ и его владельца, СИ и его терминала для аутентификации КТ с выдачей ПС билета аутентификации. Аутентификация и выпуск билета проводятся СИ и терминалом по запросу ПС при помощи КП и с согласия владельца. Непосредственную аутентификацию КТ выполняет терминал с помощью протокола BAUTH. Перед этим КТ проводит парольную аутентификацию владельца (протокол BPACE), а затем аутентификацию терминала (первая часть BAUTH).

Утверждения, которые приводятся в билете, могут быть известны СИ (например, утверждения аутентификации) или могут храниться на КТ в виде идентификационных атрибутов, и тогда терминалу требуется их получить. Перед передачей атрибутов пользователь дает согласие на передачу, а КТ проверяет, что запрос атрибутов подкреплен правами в сертификате терминала. Атрибуты пересылаются по защищенному соединению.

Определяемая схема может использоваться для построения систем массовой аутентификации. Схема может адаптироваться к нуждам систем в части организации пересылок между сторонами, поддержки дополнительных сервисов. При адаптации должна сохраняться последовательность криптографических операций схемы.

10.2 Объекты

При выпуске билета аутентификации используются следующие объекты:

- $\mathtt{Req}_{\text{ПС}}$ — запрос аутентификации;
- $H_{\text{КП}}, H_{\text{СИ}}$ — хэш-значения $\mathtt{Req}_{\text{ПС}}$, которые вычисляют КП и СИ соответственно;
- $\mathtt{Chart}_{\text{ПС}}$ — перечень утверждений, запрашиваемых ПС в $\mathtt{Req}_{\text{ПС}}$;

– $\texttt{Chart}_{\text{В}}$ — перечень идентификационных атрибутов, на предоставление которых дает согласие владелец КТ. Перечень $\texttt{Chart}_{\text{В}}$ является сужением $\texttt{Chart}_{\text{ПС}}$, которое не включает не одобренные владельцем необязательные атрибуты. Перечень кодируется словом прав доступа к прикладной программе eID. Это слово представляется объектом типа `CertHAT` (см. 7.3);

– $\texttt{Cert}(Id_T, Q_T)$ — сертификат терминала. Неявно сопровождается сертификатом корневого УЦ и может явно сопровождаться сертификатом промежуточного УЦ (см. 6.5). Маршрут сертификации определяет слово прав доступа к прикладной программе eID (см. 9.2);

– $\texttt{Attr}_{\text{КТ}}$ — идентификационные атрибуты КТ (группы данных и дополнительные атрибуты) в соответствии с перечнем $\texttt{Chart}_{\text{В}}$.

10.3 Сообщения

В таблице 6 схематически представлены сообщения, которыми обмениваются стороны при выпуске билета аутентификации.

В таблице сообщения вспомогательных протоколов BPACE и BAUTH маркируются индексом, в котором указывается название протокола. Например, M1_{BAUTH} — это сообщение M1 протокола BAUTH. Двойная стрелка ($\Rightarrow$) означает пересылку данных по защищенному соединению. Сначала это соединение «КТ — КП», затем соединение «КТ — терминал».

Формат сообщений отправляемых и возвращаемых КТ детализируется в разделе 12.

Стороны могут обмениваться дополнительными сообщениями, например, характеристиками КТ или параметрами дополнительных идентификационных атрибутов.

Таблица 6 — Выпуск билета аутентификации: сообщения

Шаг	Направление	Сообщение	Примечание
1	ПС $\rightarrow$ КП	$\texttt{Req}_{\text{ПС}}$	$\texttt{Req}_{\text{ПС}} = \langle\!\langle \texttt{Chart}_{\text{ПС}} \rangle\!\rangle$
2	КП $\rightarrow$ СИ	$\langle\!\langle \texttt{Req}_{\text{ПС}}, \texttt{Chart}_{\text{В}} \rangle\!\rangle$	
3	КП $\rightarrow$ КТ КТ $\rightarrow$ КП КП $\rightarrow$ КТ КТ $\rightarrow$ КП	$\langle\!\langle \text{M1}_{\text{BPACE}} \rangle\!\rangle$ $\langle\!\langle \text{M2}_{\text{BPACE}} \rangle\!\rangle$ $\langle\!\langle \text{M3}_{\text{BPACE}} \rangle\!\rangle$ $\langle\!\langle \text{M4}_{\text{BPACE}} \rangle\!\rangle$	$\texttt{hello}_{\text{КП}} = \langle\!\langle \texttt{Chart}_{\text{В}} \rangle\!\rangle$
4	СИ $\rightarrow$ КП	$\langle\!\langle \texttt{Cert}(Id_{\text{T}}, Q_{\text{T}}) \rangle\!\rangle$	
5	КП $\Rightarrow$ КТ КТ $\Rightarrow$ КП	$\langle\!\langle \text{M0}_{\text{BAUTH}} \rangle\!\rangle$ $\langle\!\langle \text{M1}_{\text{BAUTH}} \rangle\!\rangle$	$\text{M0}_{\text{BAUTH}} = \langle\!\langle \texttt{hello}_{\text{T}}, \texttt{Cert}(Id_{\text{T}}, Q_{\text{T}}) \rangle\!\rangle$
6	КП $\rightarrow$ СИ $\rightarrow$ Т Т $\rightarrow$ СИ $\rightarrow$ КП $\Rightarrow$ КТ КТ $\Rightarrow$ КП $\rightarrow$ СИ $\rightarrow$ Т	$\langle\!\langle \text{M1}_{\text{BAUTH}} \rangle\!\rangle$ $\langle\!\langle \text{M2}_{\text{BAUTH}} \rangle\!\rangle$ $\langle\!\langle \text{M3}_{\text{BAUTH}} \rangle\!\rangle$	
7	Т $\Rightarrow$ КТ КТ $\Rightarrow$ Т	$\langle\!\langle \texttt{Chart}_{\text{В}} \rangle\!\rangle$ $\langle\!\langle \texttt{Attr}_{\text{КТ}} \rangle\!\rangle$	По частям, через СИ и КП По частям, через КП и СИ
8	СИ $\rightarrow$ ПС	Билет	билет = $\langle\!\langle \texttt{Attr}_{\text{КТ}} \rangle\!\rangle$

10.4 Шаги

Выпуск билета аутентификации выполняется за 8 шагов, определяемых ниже. При ошибке на любом шаге, в том числе при нарушении условий проверок, выпуск билета прекращается. Правила обработки ошибок, в том числе правила информирования сторон об ошибке, в настоящем стандарте не определяются.

Шаг 1 — отправка запроса аутентификации. ПС отправляет КП запрос аутентификации $\texttt{Req}_{\text{ПС}}$. Этот запрос содержит перечень $\texttt{Chart}_{\text{ПС}}$.

Запрос должен быть волатильным: запросы одинакового содержания, выпущенные в разные моменты времени, должны отличаться. Для обеспечения волатильности в запрос может включаться отметка времени или уникальная синхропосылка.

Может требоваться, чтобы ПС подписывала запрос или включала в него аттестаты, подтверждающие права доступа к запрошенным идентификационным атрибутам. В таких случаях проверка запроса включает проверку подписи и прав доступа.

Шаг 2 — обработка запроса аутентификации. КП получает запрос $\texttt{Req}_{\text{ПС}}$, проверяет его, выделяет в запросе перечень $\texttt{Chart}_{\text{ПС}}$. Этот перечень КП согласует с владельцем КТ. Владелец выбирает устраивающие его пункты $\texttt{Chart}_{\text{ПС}}$, в результате чего формируется перечень $\texttt{Chart}_{\text{В}}$.

Затем КП вычисляет хэш-значение $H_{\text{КП}} = \texttt{belt-hash}(\texttt{Req}_{\text{ПС}})$ и отправляет СИ запрос $\texttt{Req}_{\text{ПС}}$ вместе с перечнем $\texttt{Chart}_{\text{В}}$.

СИ может быть заранее задана, либо КП может выбрать СИ из списка с участием владельца КП. Список подходящих СИ может быть дан в $\texttt{Req}_{\text{ПС}}$.

Шаг 3 — парольная аутентификация. КП и КТ выполняют протокол BPACE и формируют общий ключ K_0. Пароль протокола (PIN или CAN) передает КП владелец КТ. В качестве приветственного сообщения $\texttt{hello}_{\text{КП}}$ протокола BPACE используется перечень $\texttt{Chart}_{\text{В}}$ и возможно другие данные. КТ должен сохранить $\texttt{Chart}_{\text{В}}$ для дальнейшего использования.

КП и КТ создают защищенное соединение на ключе K_0.

Шаг 4 — выбор терминала. СИ получает запрос $\texttt{Req}_{\text{ПС}}$ и перечень $\texttt{Chart}_{\text{В}}$. СИ проверяет присланный запрос, выделяет в нем перечень $\texttt{Chart}_{\text{ПС}}$ и проверяет его соответствие $\texttt{Chart}_{\text{В}}$.

СИ выбирает терминал, права доступа которого позволяют запрашивать идентификационные атрибуты в соответствии с перечнем $\texttt{Chart}_{\text{В}}$. СИ вычисляет хэш-значение $H_{\text{СИ}} = \texttt{belt-hash}(\texttt{Req}_{\text{ПС}})$ и передает его терминалу вместе с $\texttt{Chart}_{\text{В}}$.

СИ отправляет КП сертификат $\texttt{Cert}(Id_T, Q_T)$ выбранного терминала вместе с маршрутом сертификации.

Шаг 5 — начало BAUTH. КП получает от СИ сертификат терминала. КП начинает выполнение протокола BAUTH от имени терминала, отправляя КТ сертификат $\texttt{Cert}(Id_{\text{T}}, Q_{\text{T}})$ и, при наличии, сертификат промежуточного УЦ как часть сообщения M0_{BAUTH}.

КТ проверяет полученные сертификаты и строит по маршруту сертификации слово прав доступа терминала к прикладной программе eID (см. 9.2). КТ проверяет, что терминал имеет доступ ко всем атрибутам из перечня $\texttt{Chart}_{\text{В}}$, полученного на шаге 3.

КП отправляет КТ приветственное сообщение $\texttt{hello}_{\text{T}}$ как часть сообщения M0_{BAUTH}. В $\texttt{hello}_{\text{T}}$ указывается хэш-значение $H_{\text{СИ}}$ ($H_{\text{КП}}$ на стороне КП), перечень $\texttt{Chart}_{\text{В}}$ и возможно другие данные.

КТ продолжает выполнение протокола BAUTH, обрабатывая $\texttt{hello}_{\text{T}}$. КТ формирует и отправляет КП сообщение M1_{BAUTH} с пустым приветственным сообщением $\texttt{hello}_{\text{КТ}}$.

Обмен сообщениями между КТ и КП идет по защищенному соединению, открытому на шаге 3.

КП формирует и отправляет СИ сообщение M1_{BAUTH}.

Шаг 6 — завершение BAUTH. СИ получает сообщение $M1_{BAUTH}$ и передает его терминалу.

Терминал и КТ завершают выполнение протокола BAUTH (обмениваясь сообщениями $M2_{BAUTH}$ и $M3_{BAUTH}$) и формируют общий ключ K_0. Протокол выполняется при посредничестве СИ и КП. СИ передает КП команды терминала, КП пересылает их КТ и возвращает полученные ответы СИ с последующей передачей терминалу.

Терминал и КТ создают защищенное соединение на ключе K_0. При этом КТ переключается на новое соединение с предыдущего защищенного соединения с КП.

Шаг 7 — чтение атрибутов. Терминал отправляет, а КТ получает запросы на чтение идентификационных атрибутов из перечня $\texttt{Chart}_B$. КТ проверяет, что запросы терминала соответствуют перечню, полученному на шаге 3 от КП, и возвращает запрошенные атрибуты. Терминал объединяет атрибуты в объект $\texttt{Attr}_{KT}$.

Взаимодействие терминала и КТ идет по защищенному соединению при посредничестве СИ и КП.

Примечание 1 — Защищенное соединение между КТ и терминалом может поддерживаться определенное время после аутентификации, например, для выполнения дополнительных криптографических операций.

Шаг 8 — выпуск билета аутентификации. Терминал передает СИ сформированный объект $\texttt{Attr}_{KT}$. СИ оформляет билет аутентификации и передает его ПС.

Примечание 2 — Отсутствие запрашиваемого атрибута не является ошибкой, если этот атрибут не является обязательным. При отсутствии необязательного атрибута СИ может пометить его в билете как «отсутствующий на КТ».

11 Состояния криптографического токена

КТ может находиться в следующих состояниях:

1) начальное состояние (IS);
2) состояние после успешной аутентификации по паролю (PS);
3) состояние после успешной аутентификации терминала (AS).

В состоянии IS взаимодействие с КТ осуществляется без использования защищенного соединения. В состоянии PS используется защищенное соединение «КП — КТ», а в состоянии AS — защищенные соединения «КП — КТ» и «терминал — КТ» (см. 8.5).

В состояние IS токен переходит сразу после включения или из состояний PS и AS при принудительном закрытии защищенного соединения «КП — КТ» (см. 12.4). При переходе в IS защищенное соединение «терминал — КТ», если оно было установлено ранее, закрывается. В состоянии IS доступ к прикладным программам eID и eSign запрещен.

В состояние PS токен может перейти из любого состояния после успешной парольной аутентификации, т. е. после успешного выполнения протокола BPACE, а также из состояния AS при принудительном закрытии защищенного соединения «терминал — КТ» (см. 12.4). После успешной парольной аутентификации существующее защищенное соединение «КП – КТ» переустанавливается (устанавливаются новые ключи, полученные с помощью BPACE), а существующее защищенное соединение «терминал – КТ» закрывается.

В состоянии PS может использоваться только прикладная программа eSign с правами, заданными при инициализации протокола BPACE (см. 12.2.16).

В состояние AS токен переходит из состояния PS после успешной аутентификации терминала, т. е. после успешного выполнения протокола BAUTH (с односторонней или

взаимной аутентификацией). При переходе в AS активируется защищенное соединение «терминал — КТ». В состоянии AS поддерживается переключение между защищенными соединениями «терминал — КТ» и «КП — КТ». Переключение может потребоваться, например, для подтверждения пароля PIN (см. 12.2.19), при котором ввод пароля осуществляется через КП. Состояние с активным защищенным соединением «терминал — КТ» обозначается через AS:AT, а состояние с активным защищенным соединением «КП — КТ» — через AS:CP.

В состоянии AS могут использоваться прикладные программы eSign и eID. Доступ к программам определяется правами, заданными при инициализации протокола BPACE (см. 12.2.16), и маршрутом сертификации терминала (см. 7.3).

В состояниях PS и AS все команды и соответствующие ответы должны передаваться в рамках активного защищенного соединения.

12 Командный интерфейс

12.1 Общие сведения

КП и терминал взаимодействуют с КТ с помощью команд APDU (Application Protocol Data Unit), определенных в [5]. КТ обрабатывает полученную команду, возвращая ответ. До подачи КТ новой команды должен быть получен ответ на предыдущую.

Команды и ответы на них содержат обязательные компоненты и дополнительно могут содержать необязательные.

Обязательными компонентами команды являются: CLA — класс команды, INS — инструкция команды, P1 и P2 — параметры команды. Обязательными компонентами ответа являются статусы обработки команды SW1 и SW2.

Необязательным для команды является компонент CDF, который содержит данные команды. Если компонент CDF присутствует, то команда должна также содержать необязательный компонент Lc, определяющий длину CDF.

Необязательным для ответа является компонент RDF, который содержит данные ответа. Если при выполнении команды ожидается, что в ответе будет содержаться компонент RDF, то команда должна содержать необязательный компонент Le, определяющий максимально возможную длину компонента RDF в ожидаемом ответе.

В [5] определяются соглашения о содержании компонентов команд и ответов, рассматриваются способы представления компонентов и взаимосвязь между ними. В частности, согласно [5] CLA определяет, обрабатывается ли команда как цепочка, используется ли защищенное соединение и др.

При наличии в команде компонентов Lc и Le они могут быть представлены в короткой или расширенной форме. Форма Le должна соответствовать форме Lc. Короткая и расширенная формы определяются следующим образом:

1) Lc в короткой форме состоит из одного октета, отличного от $\mathtt{00}_{16}$ и определяющего значения от 1 до 255;

2) Lc в расширенной форме состоит из трех октетов, при этом первый октет равен $\mathtt{00}_{16}$, а остальные два октета отличны от $\mathtt{0000}_{16}$ и определяют значения от 1 до 65535;

3) Le в короткой форме состоит из одного октета, определяющего значения от 1 до 256 (значению 256 соответствует $\mathtt{00}_{16}$);

4) если компонент Lc присутствует в команде, то Le в расширенной форме состоит из двух октетов, которые определяют значения от 1 до 65536 (значению 65536 соответствует $\mathtt{0000}_{16}$);

5) если компонент Lc отсутствует в команде, то Le в расширенной форме состоит из трех октетов, при этом первый октет равен $\mathtt{00}_{16}$, а следующие два октета определяют значения от 1 до 65536 (значению 65536 соответствует $\mathtt{0000}_{16}$).

В компонентах Lc и Le первый октет представляет старший байт кодируемого числа, последний октет — младший байт. Другими словами, вместо обычного для настоящего стандарта соглашения «от младших к старшим» (см. п. 4.2.2) используется соглашение «от старших к младшим» (big-endian).

В тех случаях, когда можно обойтись короткими формами Lc и Le, поддержка расширенных форм со стороны КТ не является обязательной. КП и терминал должны учитывать это при выборе форм команд в процессе взаимодействия с токеном.

В таблице 7 приводится формат пары (команда, ответ) с указанием возможных длин компонентов.

Таблица 7 — Формат пары (команда, ответ)

Компонент	Описание	Длина в октетах
CLA	Класс команды	1
INS	Инструкция команды	1
P1	Параметр команды	1
P2	Параметр команды	1
Lc	Длина данных команды	0, 1 или 3
CDF	Данные команды	0–65535
Le	Максимально возможная длина данных ответа	0–3
RDF	Данные ответа	0–65536
SW1	Статус ответа	1
SW2	Статус ответа	1

Данные в компонентах CDF и RDF либо задаются прямо, либо предварительно кодируются. При кодировании должны использоваться отличительные правила, определенные в СТБ 34.101.19 (приложение Б). В настоящем стандарте, если не оговорено противное, используются контекстно-зависимые теги, состоящие из одного или двух октетов, а данные интерпретируются как строки октетов (см. ГОСТ 34.974) длины не более 65515. При кодировании по данным $X \in \{0,1\}^{8*}$ с тегом $T \in \{0,1\}^{8*}$ строится строка октетов $\mathtt{der}(T, X) = T \parallel L \parallel X$, где $L \in \{0,1\}^{8*}$ — кодированная длина X [см. ГОСТ 34.974 (подраздел 6.3)].

Если при разборе команды или ответа обнаруживается нарушение их формата или формата вложенных кодированных данных, то разбор должен быть завершен с ошибкой.

Команды используются для реализации операций, поддерживаемых КТ. Минимальный набор операций определяется в 12.2 и кратко описывается в таблице 8. В таблице для каждой операции указываются состояния КТ, в которых операция может выполняться (см. раздел 11), и пароли, которые должны использоваться в протоколе BPACE (см. 6.3). Символ «/» в таблице обозначает «или».

Команды могут работать с файловой системой КТ, определенной в приложении Б. Файловая система имеет иерархическую структуру. Имеются назначенные файлы, которые содержат ссылки на подчиненные файлы, и элементарные файлы, в которых непосредственно размещаются объекты КТ. Выделен корневой мастер-файл, которому подчинены (прямо или по цепочке) все остальные файлы. Прикладным программам eID и eSign соответствуют специальные назначенные файлы, непосредственно подчиненные мастер-файлу.

В файловой системе всегда выбран определенный файл. Первоначально это мастер-файл. Выбор прикладных программ eID и eSign реализуется выбором соответствующих назначенных файлов. Набор разрешенных операций определяется в том числе и тем, файл какого уровня выбран в текущий момент времени: мастер-файл (MF), файл eID или подчиненный ему, файл eSign или подчиненный ему. Разрешенные уровни приводятся в последнем столбце таблицы 8.

Таблица 8 — Операции КТ

Операция	Пункт	Состояние и соединение	Пароль BPACE	Уровень
Активация личного ключа	12.2.1	PS/AS:AT	PIN	eSign
Активация PIN	12.2.2	PS/AS:AT	PUK	eID/eSign
Выбор мастер-файла	12.2.3	IS/PS/AS	любой	любой
Выбор прикладной программы	12.2.4	PS/AS	любой	любой
Выбор элементарного файла eID	12.2.5	AS:AT	CAN/PIN	eID
Выбор элементарного файла eSign	12.2.5	PS/AS:AT	PIN	eSign
Выполнение основных шагов протокола BAUTH	12.2.6	PS	любой	MF
Выполнение шагов протокола BPACE	12.2.7	IS/PS/AS:CP	—/любой	MF
Выработка подписи	12.2.8	PS/AS:AT	PIN	eSign
Генерация пары ключей	12.2.9	PS/AS:AT	PIN	eSign
Деактивация личного ключа	12.2.10	PS/AS:AT	PIN	eSign
Деактивация PIN	12.2.11	PS/AS:AT	PIN/PUK	eID/eSign
Изменение PIN	12.2.12	PS/AS:CP	PIN	eID/eSign
Инициализация выработки подписи	12.2.13	PS/AS:AT	PIN	eSign
Инициализация разбора токена ключа	12.2.14	PS/AS:AT	PIN	eSign
Инициализация протокола BAUTH	12.2.15	PS	любой	MF
Инициализация протокола BPACE	12.2.16	IS/PS/AS:CP	—/любой	MF
Обновление данных eID	12.2.17	AS:AT	CAN/PIN	eID
Обновление данных eSign	12.2.17	PS/AS:AT	PIN	eSign
Переключение между соединениями	12.2.18	AS	PIN/PUK	eID/eSign
Подтверждение PIN	12.2.19	PS/AS:CP	PIN	eSign
Проверка дополнительного атрибута	12.2.20	AS:AT	CAN/PIN	eID
Проверка сертификата	12.2.21	PS	любой	MF
Проверка флага подтверждения PIN	12.2.22	PS/AS	PIN	eSign
Разблокировка PIN	12.2.23	PS/AS:CP	PUK	eID/eSign
Разбор токена ключа	12.2.24	PS/AS:AT	PIN	eSign
Сброс флага подтверждения PIN	12.2.25	PS/AS	PIN	eSign
Чтение данных eID	12.2.26	AS:AT	CAN/PIN	eID
Чтение данных eSign	12.2.26	PS/AS:AT	PIN	eSign
Уничтожение личного ключа	12.2.27	PS/AS:AT	PIN	eSign

Типичные последовательности операций, которые могут выполняться с использованием КТ, описываются в 12.3.

В 12.4 описывается преобразование команд и ответов из исходной формы в защищенную и обратно при использовании защищенного соединения (см. 8.5).

12.2 Описание операций

12.2.1 Активация личного ключа

Для активации личного ключа используется команда <Activate>. Команда определена в таблице 9.

Таблица 9

Компонент	Описание
INS	$\texttt{44}_{16}$: активация
P1 ‖ P2	$\texttt{2100}_{16}$: активировать ключ, определяемый CDF
CDF	$\texttt{der}(\texttt{84}_{16}, X)$, где X определяет идентификатор личного ключа (см. таблицу 20)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: ключ активирован успешно
	Другое: см. таблицу В.1

При генерации личного ключа (см. 12.2.9) он автоматически активируется, и в вызове команды <Activate> нет явной необходимости. Вызов может потребоваться после принудительной деактивации ключа (см. 12.2.10).

Если ключ уже активирован, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

Команда может быть вызвана при выборе eSign в состояниях PS, AS:AT. В состоянии AS:AT команда может быть вызвана только авторизованным терминалом, в сертификате которого задано право активировать ключи (см. 7.3).

Команда требует предварительной аутентификации по PIN.

12.2.2 Активация PIN

Для активации PIN (см. 6.3) используется команда <Activate>. Команда определена в таблице 10.

Таблица 10

Компонент	Описание
INS	$\texttt{44}_{16}$: активация
P1	$\texttt{10}_{16}$: активировать пароль, определяемый P2
P2	$\texttt{03}_{16}$: PIN
CDF	—
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: PIN активирован успешно
	Другое: см. таблицу В.1

Первоначально PIN активирован, и в вызове команды <Activate> нет явной необходимости. Вызов команды может потребоваться после принудительной деактивации PIN (см. 12.2.11).

Если PIN уже активирован, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

Команда может быть вызвана при выборе eSign в состояниях PS, AS:AT и при выборе eID в состоянии AS:AT. В состоянии AS:AT команда может быть вызвана только авторизованным терминалом, в сертификате которого задано право активации пароля PIN (см. 7.3).

Команда требует предварительной аутентификации по PUK.

12.2.3 Выбор мастер-файла

Для выбора мастер-файла используется команда <Select File>. Команда определена в таблице 11.

Таблица 11

Компонент	Описание
INS	$\texttt{A4}_{16}$: выбор файла
P1 ‖ P2	$\texttt{0000}_{16}$: выбрать мастер-файл
CDF	—
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: мастер-файл выбран успешно
	Другое: см. таблицу В.1

Если мастер-файл уже выбран, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

Команда может быть вызвана в любом из состояний.

После успешного выполнения команды мастер-файл становится выбранным.

12.2.4 Выбор прикладной программы

Для выбора прикладной программы используется команда <Select File>. Команда определена в таблице 12.

Таблица 12

Компонент	Описание
INS	$\texttt{A4}_{16}$: выбор файла
P1	$\texttt{04}_{16}$: выбрать прикладную программу
P2	$\texttt{0C}_{16}$: не возвращать информацию о файле
CDF	AID прикладной программы (см. приложение Б)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: прикладная программа выбрана успешно
	$\texttt{6A82}_{16}$: файл не найден
	Другое: см. таблицу В.1

Команда может быть вызвана в состояниях PS и AS.

Для выбора прикладной программы eID требуется предварительная аутентификация по CAN, PIN или PUK.

Для выбора прикладной программы eSign требуется предварительная аутентификация по PIN или PUK.

Если прикладная программа уже выбрана, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

После успешного выполнения команды файл прикладной программы становится выбранным.

12.2.5 Выбор элементарного файла

Для выбора элементарного файла используется команда <Select File>. Команда определена в таблице 13.

Команда может вызываться в состояниях PS, AS:AT при выборе eSign и в состоянии AS:AT при выборе eID. В состоянии AS:AT команда может вызываться только авторизо-

Таблица 13

Компонент	Описание
INS	$\texttt{A4}_{16}$: выбор файла
P1	$\texttt{02}_{16}$: выбрать элементарный файл для текущей прикладной программы
P2	$\texttt{0C}_{16}$: не возвращать информацию о файле
CDF	FID файла для текущей прикладной программы (см. приложение Б)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: элементарный файл выбран успешно
	$\texttt{6A82}_{16}$: файл не найден
	Другое: см. таблицу В.1

ванным терминалом, в сертификате которого задано право доступа к соответствующему элементарному файлу (см. 7.3).

Для выбора элементарных файлов прикладной программы eSign требуется предварительная аутентификация по PIN.

Для выбора элементарных файлов прикладной программы eID требуется предварительная аутентификация по CAN или PIN.

Для выбора элементарных файлов в состоянии AS:AT требуется взаимная аутентификация КТ и терминала по протоколу BAUTH (см. 12.2.15).

После успешного выполнения команды элементарный файл становится выбранным.

Элементарные файлы, которые могут быть выбраны, приводятся в приложении Б. Выбор элементарного файла может потребоваться для чтения (см. 12.2.26) или обновления (см. 12.2.17) данных.

12.2.6 Выполнение основных шагов протокола BAUTH

Для выполнения основных шагов протокола BAUTH используется команда <General Authenticate>. Команда определена в таблице 14. Должна вызываться цепочка команд <General Authenticate> с входными и выходными данными из таблицы 15, которые определяются в соответствии с 8.4.

Таблица 14

Компонент	Описание
INS	$\texttt{86}_{16}$: общая аутентификация
P1 ‖ P2	$\texttt{0000}_{16}$: не задавать протокол (уже задан)
CDF	Отсутствует или $\texttt{der}(\texttt{7C}_{16}, X)$, где X — сообщение протокола, сформированное терминалом
RDF	Отсутствует или $\texttt{der}(\texttt{7C}_{16}, Y)$, где Y — сообщение протокола, сформированное КТ
SW1 ‖ SW2	$\texttt{9000}_{16}$: протокол (шаг) выполнен успешно
	Другое: см. таблицу В.1

Таблица 15 — Данные цепочки команд, реализующих протокол BAUTH

№ вызова	Данные в команде	Данные в ответе
1	—	$\texttt{der}(\texttt{80}_{16}, \text{M1})$
2	$\texttt{der}(\texttt{81}_{16}, \text{M2})$	$\texttt{der}(\texttt{82}_{16}, \text{M3})$

При односторонней аутентификации компонент RDF в ответе на вторую команду в цепочке не возвращается.

Команда может вызываться при выборе мастер-файла в состоянии PS непосредственно после проверки сертификата терминала (см. 12.2.21).

После успешного выполнения протокола BAUTH между КТ и терминалом устанавливается защищенное соединение (см. 12.4).

12.2.7 Выполнение шагов протокола BPACE

Для выполнения шагов протокола BPACE используется команда <General Authenticate>. Команда определена в таблице 16. Для выполнения шагов протокола BPACE должна вызываться цепочка команд <General Authenticate> с входными и выходными данными из таблицы 17, которые определяются в соответствии с 8.3.

Таблица 16

Компонент	Описание
INS	$\texttt{86}_{16}$: общая аутентификация
P1 ‖ P2	$\texttt{0000}_{16}$: не задавать протокол (уже задан)
CDF	$\texttt{der}(\texttt{7C}_{16}, X)$, где X — сообщение протокола, сформированное КП
RDF	$\texttt{der}(\texttt{7C}_{16}, Y)$, где Y — сообщение протокола, сформированное КТ
SW1 ‖ SW2	$\texttt{9000}_{16}$: протокол (шаг) выполнен успешно
	$\texttt{63CX}_{16}$: протокол (шаг) выполнен с ошибкой из-за неверного пароля, осталось X попыток аутентификации
	$\texttt{6983}_{16}$: ошибка, пароль заблокирован
	$\texttt{6984}_{16}$: ошибка, пароль деактивирован
	$\texttt{6985}_{16}$: ошибка, пароль приостановлен
	Другое: см. таблицу В.1

Статус $\texttt{63CX}_{16}$ относится ко всем паролям. Для CAN число X всегда равняется 1. Для PIN значение $\texttt{X} = 1$ означает приостановку пароля, а значение $\texttt{X} = 0$ — его блокировку. Для PUK число X равняется 9. Исключение составляет случай, когда неверный PUK последовательно предъявляется несколько раз при заблокированном PIN. В этом случае X уменьшается вплоть до 0, и при достижении нулевого значения PIN блокируется навсегда.

Последние 3 статуса ответа относятся только к паролю PIN. Статусы означают, что пароль не проверялся из-за нарушения требований к его состоянию. При статусах $\texttt{6983}_{16}$, $\texttt{6984}_{16}$ для разблокировки или активации PIN требуется выполнить BPACE с паролем PUK. При статусе $\texttt{6985}_{16}$ для возобновления PIN требуется выполнить BPACE с паролем CAN.

Таблица 17 — Данные цепочки команд, реализующих протокол BPACE

№ вызова	Данные в команде	Данные в ответе
1	$\texttt{der}(\texttt{80}_{16}, \text{M1})$	$\texttt{der}(\texttt{81}_{16}, \text{M2})$
2	$\texttt{der}(\texttt{82}_{16}, \text{M3})$	$\texttt{der}(\texttt{83}_{16}, \text{M4})$

Команда может вызываться при выборе мастер-файла в любом состоянии непосредственно после инициализации протокола BPACE (см. 12.2.16).

При успешном выполнении протокола BPACE с паролем PIN устанавливается флаг подтверждения PIN. При ошибке флаг сбрасывается.

После успешного выполнения протокола BPACE между КТ и КП устанавливается защищенное соединение (см. 12.4).

12.2.8 Выработка подписи

Для выработки подписи используется команда <PSO: Compute Digital Signature> (Perform Security Operation: Compute Digital Signature). Команда определена в таблице 18.

Таблица 18

Компонент	Описание
INS	$\texttt{2A}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{9E9A}_{16}$: выработать подпись
CDF	Хэш-значение подписываемых данных
RDF	Электронная цифровая подпись
SW1 ‖ SW2	$\texttt{9000}_{16}$: подпись выработана успешно
	Другое: см. таблицу В.1

В компоненте CDF команды должно передаваться хэш-значение подписываемых данных. Должен использоваться алгоритм хэширования, заданный при инициализации выработки подписи (см. 12.2.13). На уровне стойкости ℓ хэш-значение должно состоять из 2ℓ битов.

При выработке подписи используется алгоритм `bign-sign` (см. 8.2) со стандартными параметрами СТБ 34.101.45. Эти параметры определяются неявно по уровню стойкости личного ключа, выбранному при инициализации выработки подписи. На уровне стойкости ℓ подпись состоит из 3ℓ битов.

Команда может вызываться при выборе eSign в состояниях PS и AS:AT непосредственно после инициализации выработки подписи.

После выполнения N команд флаг подтверждения PIN сбрасывается независимо от результатов выполнения команд, где N — значение, задаваемое при инициализации протокола BPACE (см. 12.2.16). Для установки флага после сброса необходимо либо подтвердить пароль PIN (см. 12.2.19), либо повторно выполнить аутентификацию по PIN (см. 12.3.1).

12.2.9 Генерация пары ключей

Для генерации пары ключей (личного и открытого) прикладной программы eSign используется команда <Generate Asymmetric Key Pair>. При выполнении команды сгенерированный личный ключ сохраняется в KT, а открытый ключ возвращается как данные ответа. После генерации личный ключ может использоваться для выработки подписи (см. 12.2.8) и разбора токена ключа (см. 12.2.24). Команда определена в таблице 19.

Таблица 19

Компонент	Описание
INS	$\texttt{46}_{16}$: генерация пары ключей
P1 ‖ P2	$\texttt{8000}_{16}$: возвратить открытый ключ
CDF	$\texttt{der}(\texttt{B6}_{16}, \texttt{der}(\texttt{84}_{16}, X))$, где X определяет идентификатор генерируемого личного ключа (см. таблицу 20)
RDF	Открытый ключ
SW1 ‖ SW2	$\texttt{9000}_{16}$: пара ключей сгенерирована успешно
	$\texttt{6984}_{16}$: ключ уже существует
	Другое: см. таблицу В.1

Допустимые значения идентификаторов личного ключа определяются в соответствии с таблицей 20 и зависят от уровня стойкости личного ключа и состояния KT: младшая тетрада идентификатора характеризует уровень стойкости ключа, а старшая — состоя-

ние, в котором ключ может использоваться. КТ должен поддерживать генерацию ключей уровня стойкости $\ell = 128$ и может поддерживать генерацию ключей уровней стойкости $\ell = 192, 256$.

Таблица 20 — Допустимые значения идентификаторов личного ключа

Уровень стойкости ключа	Значение		Поддержка
	Состояние PS	Состояние AS	
128	$\texttt{01}_{16}$	$\texttt{11}_{16}$	Обязательна
192	$\texttt{02}_{16}$	$\texttt{12}_{16}$	Необязательна
256	$\texttt{03}_{16}$	$\texttt{13}_{16}$	Необязательна

При генерации ключей используется алгоритм `bign-sign` со стандартными параметрами СТБ 34.101.45, которые определяются неявно по уровню стойкости личного ключа. На уровне стойкости ℓ открытый ключ задается 4ℓ битами по правилам, установленным в СТБ 34.101.45.

Команда может вызываться при выборе eSign в состояниях PS и AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право генерации ключей в терминальном режиме (см. 7.3).

Команда требует предварительной аутентификации по PIN, а также установки флага подтверждения PIN.

Выполнение команды приводит к сбросу флага подтверждения PIN. Для установки флага после сброса необходимо либо подтвердить пароль PIN (см. 12.2.19), либо повторно выполнить аутентификацию по PIN (см. 12.3.1).

Если при выполнении команды личный ключ с указанным в команде идентификатором уже существует, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{6984}_{16}$. При этом для генерации нового ключа старый ключ должен быть предварительно уничтожен командой <Terminate> (см. 12.2.27).

12.2.10 Деактивация личного ключа

Для деактивации личного ключа используется команда <Deactivate>. Команда определена в таблице 21.

Таблица 21

Компонент	Описание
INS	$\texttt{04}_{16}$: деактивация
P1 ‖ P2	$\texttt{2100}_{16}$: деактивировать ключ, определяемый CDF
CDF	$\texttt{der}(\texttt{84}_{16}, X)$, где X определяет идентификатор личного ключа (см. таблицу 20)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: ключ деактивирован успешно
	Другое: см. таблицу В.1

Команда может вызываться при выборе eSign в состояниях PS и AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право деактивировать ключи (см. 7.3).

Команда требует предварительной аутентификации по PIN.

Если ключ уже деактивирован, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

После деактивации личного ключа выполнение любых операций, требующих его использования, становится невозможным.

12.2.11 Деактивация PIN

Для деактивации PIN (см. 6.3) используется команда <Deactivate>. Команда определена в таблице 22.

Таблица 22

Компонент	Описание
INS	$\texttt{04}_{16}$: деактивация
P1	$\texttt{10}_{16}$: деактивировать пароль, определяемый P2
P2	$\texttt{03}_{16}$: PIN
CDF	—
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: PIN деактивирован успешно
	Другое: см. таблицу B.1

Команда может вызываться при выборе eSign в состояниях PS, AS:AT и при выборе eID в состоянии AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право деактивировать пароль PIN (см. 7.3).

Команда требует предварительной аутентификации по PIN или PUK.

Если PIN уже деактивирован, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

После деактивации PIN выполнение любых операций, требующих аутентификации по паролю PIN (см. таблицу 8), становится невозможным. При этом KT переходит в состояние IS и мастер-файл становится выбранным файлом.

12.2.12 Изменение PIN

Для изменения PIN используется команда <Change reference data>. Команда определена в таблице 23.

Таблица 23

Компонент	Описание
INS	$\texttt{24}_{16}$: изменение данные
P1	$\texttt{00}_{16}$: задать старое и новое значения пароля
P2	$\texttt{03}_{16}$: PIN
CDF	Старое и новое значения PIN (6 + 6 октетов)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: PIN изменен успешно
	Другое: см. таблицу B.1

Команда может вызываться при выборе eSign в состояниях PS, AS:CP и при выборе eID в состоянии AS:CP. В состоянии AS:CP команда может вызываться только в случае, если в сертификате авторизованного терминала задано право изменения пароля PIN (см. 7.3).

Команда требует предварительной аутентификации по PIN.

Выполнение команды приводит к сбросу флага подтверждения PIN. Для установки флага после сброса необходимо либо подтвердить пароль PIN (см. 12.2.19), либо повторно выполнить аутентификацию по PIN (см. 12.3.1).

12.2.13 Инициализация выработки подписи

Для инициализации выработки подписи используется команда <MSE: Set DST> (Manage Security Environment: Set Digital Signature Template). Команда задает личный ключ и алгоритм хэширования, которые будут использоваться при выработке подписи. Команда определена в таблице 24.

Таблица 24

Компонент	Описание
INS	$\texttt{22}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{41B6}_{16}$: инициализировать алгоритм выработки подписи
CDF	$\texttt{der}(\texttt{84}_{16}, X)$ ‖ $\texttt{der}(\texttt{80}_{16}, Y)$, где X определяет идентификатор личного ключа (см. таблицу 20), а Y — идентификатор используемого при выработке подписи алгоритма хэширования и принимает значение $\texttt{90}_{16}$ для алгоритма хэширования `belt-hash`, $\texttt{B0}_{16}$ для алгоритма хэширования `bash384` и $\texttt{C0}_{16}$ для алгоритма хэширования `bash512` (см. 8.2)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: алгоритм инициализирован успешно
	$\texttt{6283}_{16}$: личный ключ деактивирован
	$\texttt{6984}_{16}$: личный ключ уничтожен
	Другое: см. таблицу В.1

В компоненте CDF алгоритм хэширования `belt-hash` может быть задан только совместно с личным ключом уровня стойкости $\ell = 128$, а алгоритмы хэширования `bash384` и `bash512` — с личными ключами уровней стойкости $\ell = 192$ или $\ell = 256$ соответственно. Уровень стойкости алгоритма хэширования `bash` определяется неявно по уровню стойкости выбранного личного ключа.

Команда требует предварительной аутентификации по PIN, а также установки флага подтверждения PIN. Для установки флага, если он был сброшен, необходимо либо подтвердить пароль PIN (см. 12.2.19), либо повторно выполнить аутентификацию по паролю PIN (см. 12.3.1).

Команда может вызываться при выборе eSign в состояниях PS и AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право выработки подписи в терминальном режиме (см. 7.3).

12.2.14 Инициализация разбора токена ключа

Для инициализации разбора токена ключа используется команда <MSE: Set CT> (Manage Security Environment: Set Confidentialy template). Команда задает личный ключ, который будет использоваться при разборе токена ключа. Команда определена в таблице 25.

Команда может вызываться при выборе eSign в состояниях PS и AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право разбора токена ключа в терминальном режиме (см. 7.3).

Команда требует предварительной аутентификации по PIN.

12.2.15 Инициализация протокола BAUTH

Для инициализации протокола BAUTH используется команда <MSE: Set AT> (Manage Security Environment: Set Authentication Template). Команда определена в таблице 26.

Таблица 25

Компонент	Описание
INS	$\texttt{22}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{41B8}_{16}$: инициализировать алгоритм разбора токена ключа (расшифрования ключа)
CDF	$\texttt{der}(\texttt{84}_{16}, X)$, где X определяет идентификатор личного ключа (см. таблицу 20)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: алгоритм инициализирован успешно
	$\texttt{6283}_{16}$: личный ключ деактивирован
	$\texttt{6984}_{16}$: личный ключ уничтожен
	Другое: см. таблицу В.1

В компоненте CDF обязательным является только первый объект. Объекты должны передаваться с помощью одной команды, т. е. использование цепочки команд <MSE: Set AT> не допускается.

Команда может вызываться при выборе мастер-файла в состоянии PS непосредственно после выполнения протокола BPACE (см. 12.2.7). Приветственное сообщение $\texttt{hello}_{\text{T}}$ протокола BAUTH определяется как CDF ‖ $\texttt{CertHAT}^*$. Здесь $\texttt{CertHAT}^*$ — список объектов CertHAT, указанный ранее в команде инициализации протокола BPACE (см. 12.2.16). Приветственное сообщение $\texttt{hello}_{\text{КТ}}$ протокола BAUTH полагается пустым.

При успешном выполнении протокола BAUTH дата, которая передается в команде для проверки срока действия КТ, может использоваться для обновления даты, которая хранится на КТ (см. 6.2).

12.2.16 Инициализация протокола BPACE

Для инициализации протокола BPACE используется команда <MSE: Set AT> (Manage Security Environment: Set Authentication Template). Команда определена в таблице 27.

Тип пароля, который должен использоваться при инициализации BPACE для выполнения тех или иных операций, определяется в таблице 8.

Последние 4 статуса ответа (кроме $\texttt{9000}_{16}$) относятся только к паролю PIN. При статусах $\texttt{6983}_{16}$, $\texttt{6984}_{16}$ для разблокировки или активации PIN требуется выполнить BPACE с паролем PUK. При статусе $\texttt{6985}_{16}$ для возобновления PIN требуется выполнить BPACE с паролем CAN.

Все объекты данных, которые требуется передать через команду <MSE: Set AT>, должны укладываться в один компонент CDF и передаваться за один вызов команды. Другими словами, использование цепочки команд <MSE: Set AT> не допускается. При этом порядок следования объектов данных в CDF не важен. Например, для протокола BPACE, использующего PIN, компонент CDF, содержащий только обязательные объекты данных, может иметь вид: CDF = $\texttt{der}(\texttt{83}_{16}, \texttt{03}_{16})$ ‖ $\texttt{der}(\texttt{80}_{16}, X)$. Здесь X — закодированный (без поля тега и поля длины) объектный идентификатор протокола BPACE (см. СТБ 34.101.66).

Компонент CDF должен использоваться в протоколе BPACE в качестве приветственного сообщения $\texttt{hello}_{\text{КП}}$ (см. 8.3).

Команда может вызываться в любом из состояний при выборе мастер-файла.

Таблица 26

Компонент	Описание
INS	$\texttt{22}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{F1A4}_{16}$: выбрать и инициализировать протокол BAUTH с взаимной аутентификацией
	$\texttt{B1A4}_{16}$: выбрать и инициализировать протокол BAUTH с односторонней аутентификацией
CDF	Объект данных $\texttt{der}(\texttt{80}_{16}, X)$, где X — закодированный объектный идентификатор (без поля тега и поля длины) протокола (см. приложение А). Объект данных $\texttt{der}(\texttt{85}_{16}, X)$, где X определяет хэш-значение запроса аутентификации (см. раздел 10). Объект данных X, который является закодированным значением типа `AuthAuxData` (см. 7.2). Используется в команде <Verify> (см. 12.2.20) для получения значений дополнительных атрибутов DocumentValidity, AgeVerification, RegionVerification. В X обязательно должны быть включены параметры атрибута DocumentValidity.
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: протокол инициализирован успешно
	Другое: см. таблицу В.1

При успешном выполнении команды защищенное соединение «терминал — КТ», если оно было установлено ранее, закрывается.

12.2.17 Обновление данных

Для обновления данных элементарных файлов используется команда <Update Binary>. Команда определена в таблице 28.

Размер данных, которые нужно записать, определяется компонентом Lc команды (см. [5]).

Команда может вызываться в состояниях PS, AS:AT при выборе eSign и в состоянии AS:AT при выборе eID. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право записи в соответствующий элементарный файл (см. 7.3).

Для обновления элементарных файлов прикладной программы eSign требуется предварительная аутентификация по PIN.

Для обновления элементарных файлов прикладной программы eID требуется предварительная аутентификация по CAN или PIN.

Файл, в который записываются данные, должен быть предварительно выбран (см. 12.2.5). Запись может быть произведена только в те файлы, для которых нет ограничений по записи при текущем состоянии КТ. При попытке записи в файлы, доступ к которым ограничен, должен возвращаться статус SW1 ‖ SW2 = $\texttt{6982}_{16}$.

12.2.18 Переключение между соединениями

Для переключения между защищенным соединением, устанавливаемым между КТ и КП после выполнения протокола BPACE, и соединением, устанавливаемым между КТ и терминалом после выполнения протокола BAUTH, используется команда <MSE: Set CS>

Таблица 27

Компонент	Описание
INS	$\texttt{22}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{C1A4}_{16}$: выбрать и инициализировать протокол BPACE
CDF	Обязательный объект данных $\texttt{der}(\texttt{80}_{16}, X)$, где X — закодированный объектный идентификатор (без поля тега и поля длины) протокола (см. приложение А). Обязательный объект данных $\texttt{der}(\texttt{83}_{16}, X)$, где X определяет, какой пароль будет использоваться в протоколе: $\texttt{02}_{16}$ — CAN, $\texttt{03}_{16}$ — PIN, $\texttt{04}_{16}$ — PUK. Необязательные объекты данных типа `CertHAT` (см. 7.3). Используются для установки прав доступа к данным и сервисам прикладных программ КТ. Для каждой прикладной программы используется свой объект. Отсутствие объекта для определенной прикладной программы означает отсутствие прав доступа к ней. Необязательный объект данных $\texttt{der}(\texttt{53}_{16}, X)$, где X принимает значения от $\texttt{00}_{16}$ до $\texttt{FF}_{16}$. Октет X кодирует число N последовательных подписей, которые могут быть выработаны прикладной программой eSign без повторного подтверждения PIN. Значение $N = 0$ означает, что может быть выработано неограниченное количество подписей. Если объект $\texttt{der}(\texttt{53}_{16}, X)$ не задан, то eSign использует $N = 1$
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: протокол инициализирован успешно, число возможных попыток аутентификации равно максимальному значению
	$\texttt{63CX}_{16}$: протокол инициализирован успешно, осталось `X` попыток аутентификации и число попыток меньше максимального
	$\texttt{6983}_{16}$: ошибка, пароль заблокирован
	$\texttt{6984}_{16}$: ошибка, пароль деактивирован
	$\texttt{6985}_{16}$: ошибка, пароль приостановлен
	Другое: см. таблицу В.1

(аббревиатура от «Manage Security Environment: Set Context Switch>). Команда определена в таблице 29.

Команда может вызываться при выборе eID или eSign в состоянии AS.

Команда требует предварительной аутентификации по PIN или PUK.

Переключение между соединениями может понадобиться при подтверждении (см. 12.2.19), изменении (см. 12.2.12) или разблокировке (см. 12.2.23) PIN.

После успешного выполнения команды мастер-файл становится выбранным.

12.2.19 Подтверждение PIN

Для подтверждения PIN используется команда <Verify>. Команда определена в таблице 30.

Команда может вызываться при выборе eSign в состояниях PS и AS:CP.

Команда требует предварительной аутентификации по PIN.

При успешном выполнении команды устанавливается флаг подтверждения PIN. При ошибке флаг сбрасывается.

Таблица 28

Компонент	Описание
INS	$\texttt{D6}_{16}$: обновление двоичных данных
P1	Старший байт смещения, с которого будут перезаписываться данные (старший бит должен быть равен нулю)
P2	Младший байт смещения, с которого будут перезаписываться данные
CDF	Записываемые данные
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: данные перезаписаны успешно
	Другое: см. таблицу В.1

Таблица 29

Компонент	Описание
INS	$\texttt{22}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{01A4}_{16}$: переключиться между соединениями
CDF	$\texttt{der}(\texttt{E1}_{16}, \texttt{der}(\texttt{81}_{16}, X))$, где X определяет идентификатор соединения и принимает значение $\texttt{00}_{16}$ для переключения на защищенное соединение «КП – КТ» и значение $\texttt{01}_{16}$ для переключения на защищенное соединение «терминал – КТ»
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: переключение между соединениями выполнено успешно
	Другое: см. таблицу В.1

Вызов команды может потребоваться после сброса флага подтверждения PIN, принудительного (см. 12.2.25) или автоматического, которое происходит после выработки определенного количества подписей (см. 12.2.8), генерации пары ключей (см. 12.2.9), изменения пароля PIN (см. 12.2.12), уничтожения личного ключа (см. 12.2.27).

12.2.20 Проверка дополнительного атрибута

Для проверки дополнительных атрибутов DocumentValidity, AgeVerification, RegionVerification (см. 7.2) используется команда <Verify>. Команда определена в таблице 31.

Таблица 30

Компонент	Описание
INS	$\texttt{20}_{16}$: проверка данных
P1 ‖ P2	$\texttt{0003}_{16}$: проверить пароль PIN
CDF	Пароль PIN (6 октетов)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: аутентификация успешна
	$\texttt{63CX}_{16}$: аутентификация прошла с ошибкой, значение $\texttt{X}$ содержит количество оставшихся попыток аутентификации (при $\texttt{X} = 1$ пароль приостановлен, а при $\texttt{X} = 0$ — заблокирован)
	$\texttt{6984}_{16}$: пароль деактивирован
	Другое: см. таблицу В.1

Таблица 31

Компонент	Описание
INS	$\texttt{20}_{16}$: проверка данных
P1 ‖ P2	$\texttt{8000}_{16}$: проверить дополнительный атрибут
CDF	Закодированный идентификатор дополнительного атрибута (см. 7.2). Может принимать одно из значений `id-DocumentValidity`, `id-AgeVerification` или `id-PlaceVerification` (см. приложение А)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: атрибут проверен успешно
	$\texttt{6200}_{16}$: ошибка при проверке атрибута
	Другое: см. таблицу В.1

Команда может вызываться при выборе eID в состоянии AS:AT авторизованным терминалом, в сертификате которого задано право проверки соответствующего атрибута (см. 7.3).

Команда требует предварительной аутентификации по CAN или PIN.

Установка проверяемых командой данных производится при инициализации протокола BAUTH (см. 12.2.15).

Для команды должен использоваться CLA = $\texttt{84}_{16}$ (прикладной класс для защищенного соединения без использования цепочки команд).

12.2.21 Проверка сертификата

Для импорта и проверки сертификата при аутентификации терминала по протоколу BAUTH используется команда <PSO: Verify Certificate> (Perform Security Operation: Verify Certificate). Команда определена в таблице 32.

Таблица 32

Компонент	Описание
INS	$\texttt{2A}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{00BE}_{16}$: проверить сертификат открытого ключа
CDF	Закодированное значение компонента `certificateBody`, определяющего тело облегченного сертификата (см. 9.1). Закодированное значение компонента `signature`, определяющего подпись облегченного сертификата (см. 9.1)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: сертификат проверен успешно
	Другое: см. таблицу В.1

В компоненте CDF команды передаются тело и подпись облегченного сертификата. Для их передачи может использоваться цепочка из двух команд <PSO: Verify Certificate>, первая из которых содержит тело облегченного сертификата, а вторая — подпись облегченного сертификата.

Для передачи нескольких сертификатов, составляющих маршрут сертификации, должен использоваться последовательный вызов команд <PSO: Verify Certificate>. При этом если какой-либо сертификат передается цепочкой команд, то для него должна использоваться своя цепочка.

Открытый ключ, используемый при проверке сертификата после его передачи в КТ, извлекается из сертификата, который хранится на КТ или который был импортирован в КТ предшествующим успешным вызовом команды <PSO: Verify Certificate> (см. 9.2).

Команда может вызываться при выборе мастер-файла в состоянии PS непосредственно после инициализации протокола BAUTH (см. 12.2.15).

Команда требует предварительной аутентификации по CAN, PIN или PUK. Если аутентификация выполнялась по CAN, то в сертификате терминала должно быть установлено право доступа по паролю CAN (см. 7.3).

Если сертификат является недействительным, то должен возвращаться статус SW1 ‖ SW2 = $\texttt{6A80}_{16}$.

12.2.22 Проверка флага подтверждения PIN

Для проверки флага подтверждения PIN используется команда <Verify>. Команда определена в таблице 33.

Таблица 33

Компонент	Описание
INS	$\texttt{20}_{16}$: проверка данных
P1 ‖ P2	$\texttt{0003}_{16}$: проверить флаг подтверждения PIN
CDF	—
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: флаг подтверждения PIN установлен
	Другое: см. таблицу В.1

Команда может вызываться при выборе eSign в состояниях PS, AS:CP и AS:AT.

Команда требует предварительной аутентификации по PIN.

Если флаг подтверждения PIN не установлен, то должен возвращаться статус SW1 ‖ SW2 = $\texttt{6982}_{16}$.

12.2.23 Разблокировка PIN

Для разблокировки PIN используется команда <Reset Retry Counter>. Команда определена в таблице 34.

Таблица 34

Компонент	Описание
INS	$\texttt{2C}_{16}$: разблокировка PIN
P1	$\texttt{02}_{16}$: разблокировать PIN с его изменением
P1	$\texttt{03}_{16}$: разблокировать PIN без его изменения
P2	$\texttt{00}_{16}$: PIN выбран неявно
CDF	При P1 = $\texttt{02}_{16}$ новое значение PIN (6 октетов)
	При P1 = $\texttt{03}_{16}$ не задается
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: разблокировка PIN была выполнена успешно
	Другое: произошла ошибка (см. таблицу В.1)

Команда может вызываться при выборе eSign в состояниях PS, AS:CP и при выборе eID в состоянии AS:CP. В состоянии AS:CP команда может вызываться только в случае, если в сертификате авторизованного терминала задано право разблокировать пароль PIN (см. 7.3).

Команда требует предварительной аутентификации по PUK.

12.2.24 Разбор токена ключа

Для разбора токена ключа используется команда <PSO: Decipher> (Perform Security Operation: Decipher). Команда определена в таблице 35.

Таблица 35

Компонент	Описание
INS	$\texttt{2A}_{16}$: управление средой безопасности
P1 ‖ P2	$\texttt{8086}_{16}$: расшифровать данные
CDF	Объект $I \parallel Y$, где I — заголовок транспортируемого ключа, Y — токен транспортируемого ключа
RDF	Расшифрованный транспортируемый ключ
SW1 ‖ SW2	$\texttt{9000}_{16}$: токен ключа разобран успешно
	Другое: см. таблицу В.1

При разборе токена ключа используется алгоритм $\texttt{bign-keytransport}^{-1}$ (см. 8.2) со стандартными параметрами СТБ 34.101.45. Эти параметры определяются неявно по уровню стойкости личного ключа, выбранному при инициализации разбора токена (см. 12.2.14).

Битовая длина транспортируемого ключа, который определяется при разборе токена, должна принимать одно из трех значений — $k = 128$, $k = 192$ или $k = 256$. При этом длина k не должна быть больше уровня стойкости ℓ личного ключа, используемого для разбора. Входной токен Y должен состоять из $k + 2\ell + 128$ битов, заголовок I является 128-битовым.

Команда может вызываться при выборе eSign в состояниях PS и AS:AT непосредственно после успешной инициализации разбора токена (см. 12.2.14).

12.2.25 Сброс флага подтверждения PIN

Для сброса флага подтверждения PIN используется команда <Verify>. Команда определена в таблице 36.

Таблица 36

Компонент	Описание
INS	$\texttt{20}_{16}$: проверка данных
P1 ‖ P2	$\texttt{FF03}_{16}$: сбросить флаг подтверждения PIN
CDF	—
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$ флаг подтверждения PIN сброшен успешно
	Другое: см. таблицу В.1

Команда может вызываться при выборе eSign в состояниях PS, AS:CP и AS:AT.

Команда требует предварительной аутентификации по PIN.

При успешном выполнении команды флаг подтверждения PIN сбрасывается. При ошибке флаг не изменяется.

Если флаг подтверждения PIN уже сброшен, то должен быть возвращен статус SW1 ‖ SW2 = $\texttt{9000}_{16}$.

12.2.26 Чтение данных

Для чтения данных из элементарных файлов используется команда <Read Binary>. Команда определена в таблице 37.

Таблица 37

Компонент	Описание
INS	$\texttt{B0}_{16}$: чтение двоичных данных
P1	Старший байт смещения, с которого будут прочитываться данные (старший бит должен быть равен нулю)
P2	Младший байт смещения, с которого будут прочитываться данные
CDF	—
RDF	Прочитанные данные
SW1 ‖ SW2	$\texttt{9000}_{16}$: данные прочитаны успешно
	Другое: см. таблицу В.1

Размер данных, которые нужно прочитать, определяется компонентом Le (см. [5]).

Команда может вызываться в состояниях PS, AS:AT для файлов eSign и в состоянии AS:AT для файлов eID. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право чтения соответствующего файла или группы данных (см. 7.3).

Для чтения элементарных файлов прикладной программы eSign требуется предварительная аутентификация по PIN.

Для чтения элементарных файлов прикладной программы eID требуется предварительная аутентификация по CAN или PIN.

Элементарный файл, из которого прочитываются данные, должен быть предварительно выбран (см. 12.2.5). Прочитаны могут быть только те данные, для которых нет ограничений по чтению при текущем состоянии КТ (см. 7.3). При попытке чтения данных, доступ к которым ограничен, должен возвращаться статус SW1 ‖ SW2 = $\texttt{6982}_{16}$.

12.2.27 Уничтожение личного ключа

Для уничтожения личного ключа используется команда <Terminate>. Команда определена в таблице 38.

Таблица 38

Компонент	Описание
INS	$\texttt{E6}_{16}$: уничтожение личного ключа
P1 ‖ P2	$\texttt{2100}_{16}$: использовать идентификатор личного ключа из поля данных
CDF	$\texttt{der}(\texttt{B6}_{16}, \texttt{der}(\texttt{84}_{16}, X))$, где X определяет идентификатор личного ключа (см. таблицу 20)
RDF	—
SW1 ‖ SW2	$\texttt{9000}_{16}$: ключ уничтожен успешно
	Другое: см. таблицу В.1

Команда может вызываться при выборе eSign в состояниях PS и AS:AT. В состоянии AS:AT команда может вызываться только авторизованным терминалом, в сертификате которого задано право уничтожать ключи (см. 7.3). В каждом из состояний могут быть уничтожены только те ключи, которые были сгенерированы в данном состоянии (см. 12.2.9).

Команда требует предварительной аутентификации по PIN. При вызове команды флаг подтверждения PIN должен быть установлен.

Выполнение команды приводит к сбросу флага подтверждения PIN. Для установки флага после сброса необходимо либо подтвердить пароль PIN (см. 12.2.19), либо повторно выполнить аутентификацию по PIN (см. 12.3.1).

При попытке уничтожить ключ, который не был сгенерирован или который был уничтожен ранее, должен возвращаться статус SW1 $\|$ SW2 $= \mathtt{9000}_{16}$.

12.3 Последовательности операций

12.3.1 Аутентификация по паролю

Для аутентификации по паролю используется протокол BPACE (см. 8.3). Для аутентификации на стороне КТ должна использоваться следующая последовательность операций:

1 Выбрать мастер-файл (см. 12.2.3), если он не является текущим назначенным файлом.

2 Инициализировать протокол BPACE (см. 12.2.16).

3 Выполнить шаги протокола BPACE (см. 12.2.7).

При аутентификации по паролю могут устанавливаться ограничения на доступ к данным и операциям КТ. Данные ограничения задаются при инициализации протокола BPACE (см. 12.2.16).

При первом выполнении протокола BPACE обмен данными КП с КТ производится по незащищенному соединению. При успешном выполнении протокола между КТ и КП устанавливается защищенное соединение и КТ переходит в состояние PS (см. раздел 11). Новый сеанс протокола (если он предусмотрен) будет выполняться уже по защищенному соединению.

Успешная аутентификация по паролю CAN возобновляет приостановленный пароль PIN (см. 6.3).

12.3.2 Аутентификация терминала и КТ

Аутентификация терминала и КТ производится по протоколу BAUTH (см. 8.4) после успешной аутентификации по паролю (см. 12.3.1). Аутентификация может быть односторонней (аутентификация терминала перед КТ) или взаимной (аутентификация терминала перед КТ и КТ перед терминалом). Для выполнения аутентификации должна использоваться следующая последовательность операций:

1 Инициализировать протокол BAUTH (см. 12.2.15).

2 Проверить сертификат терминала (см. 12.2.21).

3 Выполнить основные шаги протокола BAUTH (см. 12.2.6).

После успешного выполнения протокола BAUTH с взаимной аутентификацией сторон срок действия КТ может быть проверен с использованием следующей последовательности операций:

1 Выбрать прикладную программу eID (см. 12.2.4).

2 Проверить дополнительный атрибут DocumentValidity (см. 12.2.20).

При выполнении протокола BAUTH обмен данными терминала с КТ производится по защищенному соединению, установленному после выполнения протокола BPACE. При успешном выполнении протокола BAUTH между терминалом и КТ создается новое защищенное соединение, которое устанавливается в качестве текущего соединения, при этом КТ переходит в состояние AS (см. раздел 11).

12.3.3 Активация пароля PIN

Активация пароля PIN может потребоваться после его принудительной деактивации (см. 12.2.11).

Для активации пароля PIN без использования терминала должна использоваться следующая последовательность операций:

1 Выполнить аутентификацию по паролю PUK (см. 12.3.1).

2 Выбрать прикладную программу eSign (см. 12.2.4).

3 Активировать пароль PIN (см. 12.2.2).

Для активации пароля PIN с использованием терминала должна использоваться следующая последовательность операций:

1 Выполнить аутентификацию по паролю PUK (см. 12.3.1).

2 Выполнить аутентификацию терминала (см. 12.3.2), в сертификате которого задано право активировать пароль PIN (см. 7.3).

3 Выбрать прикладную программу (см. 12.2.4), для которой в сертификате авторизованного терминала установлено право активировать пароль PIN (см. 7.3).

4 Активировать пароль PIN (см. 12.2.2).

Активация пароля PIN не приводит к изменению состояния КТ. После активации пароля PIN для получения доступа к операциям, которые требуют предварительной аутентификации по данному паролю, необходимо выполнить аутентификацию по паролю PIN (см. 12.3.1) и, при необходимости, аутентификацию терминала и КТ (см. 12.3.2).

12.3.4 Управление паролем PIN

К операциям по управлению паролем PIN относятся подтверждение (см. 12.2.19), изменение (см. 12.2.12) и разблокировка (см. 12.2.23).

Для подтверждения или изменения пароля PIN предварительно должна быть выполнена аутентификация по паролю PIN, а для разблокировки пароля PIN — аутентификация по паролю PUK (см. 12.3.1). Разблокировка может выполняться с изменением или без изменения пароля PIN (см. 12.2.23).

После успешной аутентификации по паролю, при необходимости, может быть выполнена аутентификация терминала и КТ (см. 12.3.2), при этом в сертификате терминала должно быть задано соответствующее право по управлению паролем PIN (см. 7.3).

Изменение или разблокировка PIN может производится прикладной программой eID или eSign, а подтверждение пароля PIN — прикладной программой eSign. Используемая для управления паролем PIN прикладная программа должна быть предварительно выбрана (см. 12.2.4).

В состоянии PS или AS:CP для управления паролем PIN достаточно выполнить лишь соответствующую операцию.

В состоянии AS:AT для управления паролем PIN может использоваться следующая последовательность операций:

1 Переключиться на защищенное соединение «КП — КТ» (см. 12.2.18).

2 Выбрать прикладную программу (см. 12.2.4), для которой в сертификате авторизованного терминала задано соответствующее право по управлению паролем PIN (см. 7.3).

3 Выполнить требуемую операцию по управлению паролем PIN.

4 Переключиться на защищенное соединение «терминал — КТ» (см. 12.2.18).

5 При необходимости выбрать нужную прикладную программу (см. 12.2.4).

12.3.5 Генерация пары ключей и установка сертификата

Для генерации пары ключей и установки сертификата предварительно должна быть выполнена аутентификация по паролю PIN (см. 12.3.1).

После успешной аутентификации по паролю, при необходимости, может быть выполнена взаимная аутентификация терминала и КТ (см. 12.3.2), при этом в сертификате терминала должно быть задано право генерации ключей (см. 7.3).

Генерация пары ключей и установка сертификата производятся прикладной программой eSign, которая должна быть предварительно выбрана (см. 12.2.4).

В состояниях PS и AS может быть сгенерировано до трех ключевых пар (см. 12.2.9), соответствующих различным уровням стойкости. Сгенерированные личные ключи могут использоваться для подписи данных (см. 12.3.6) и разбора токена ключа (см. 12.3.7). При этом личный ключ, сгенерированный в одном из состояний, не может использоваться в другом состоянии.

Открытый ключ, который возвращается при генерации пары ключей (см. 12.2.9), должен использоваться при формировании запроса на получение сертификата. При формировании запроса дополнительно может использоваться объект `Name` (см. 6.6). Запрос на получение сертификата должен быть подписан (см. 12.3.6) личным ключом, соответствующим открытому ключу из запроса.

Выпущенный сертификат может быть записан на КТ. Для записи сертификата должна использоваться операция обновления данных (см. 12.2.17). В состоянии AS запись сертификата должна выполняться авторизованным терминалом, в сертификате которого задано соответствующее право (см. 7.3).

После формирование запроса и до записи сертификата рекомендуется хранить на месте сертификата хэш-значение запроса, вычисленное с помощью алгоритма `belt-hash`. Хэш-значение может быть использовано для получения от УЦ сертификата после его выпуска (см. СТБ 34.101.78).

Сертификаты и объект `Name` хранятся на КТ в определенных элементарных файлах, описанных в таблице Б.2.

12.3.6 Выработка подписи

Для выработки подписи предварительно должна быть выполнена аутентификация по паролю PIN (см. 12.3.1).

После успешной аутентификации по паролю, при необходимости, может быть выполнена аутентификация терминала и КТ (см. 12.3.2); при этом в сертификате терминала должно быть задано право вырабатывать подпись (см. 7.3).

Выработка подписи производится прикладной программой eSign, которая должна быть предварительно выбрана (см. 12.2.4).

Для выработки подписи должна использоваться следующая последовательность операций:

1 Если флаг подтверждения PIN не установлен, подтвердить пароль PIN (см. 12.3.4).

2 Инициализировать алгоритм выработки подписи (см. 12.2.13).

3 Выработать подпись (см. 12.2.8).

Указанная последовательность операций должна выполняться при каждой выработке подписи.

Для проверки флага подтверждения PIN может использоваться соответствующая операция (см. 12.2.22).

12.3.7 Разбор токена ключа

Для разбора токена ключа предварительно должна быть выполнена аутентификация по паролю PIN (см. 12.3.1).

После успешной аутентификации по паролю, при необходимости, может быть выполнена аутентификация терминала и КТ (см. 12.3.2), при этом в сертификате терминала должно быть задано право разбирать токен ключа (см. 7.3).

Разбор токена ключа производится прикладной программой eSign, которая должна быть предварительно выбрана (см. 12.2.4).

Для разбора токена ключа должна использоваться следующая последовательность операций:

1 Инициализировать алгоритм разбора токена ключа (см. 12.2.14).

2 Разобрать токен ключа (см. 12.2.24).

Указанная последовательность операций должна выполняться при каждом разборе токена ключа.

12.4 Управление защищенным соединением

12.4.1 Форматы сообщений

Команда и ответ на команду передаются в виде двоичных слов cmd = CLA ‖ INS ‖ P1 ‖ P2 ‖ Lc ‖ CDF ‖ Le и res = RDF ‖ SW1 ‖ SW2 соответственно (см. таблицу 7). Компоненты Lc, CDF, Le и RDF могут быть пустыми словами, т. е. отсутствовать.

При передаче по защищенному соединению команда cmd преобразуются в защищенную команду cmd* = CLA* ‖ INS ‖ P1 ‖ P2 ‖ Lc* ‖ CDF* ‖ Le*, а ответ res — в защищенный ответ res* = RDF* ‖ SW1 ‖ SW2. В защищенных командах и ответах все компоненты имеют ненулевую длину.

Компонент CLA* защищенной команды получается из CLA установкой признака (бита) защиты, Lc* кодирует длину компонента CDF*, а Le* всегда устанавливается в $\texttt{00}_{16}$. В CDF* включаются зашифрованный компонент CDF (если он непуст), компонент Le (если непуст) и имитовставка (обязательно).

Аналогично компонент RDF* защищенного ответа включает зашифрованный компонент RDF (если он непуст) и имитовставку (обязательно).

При формировании компонентов CDF* и RDF* включаемые в них объекты данных кодируются с использованием отличительных правил (см. 12.1). В таблице 39 приводятся допустимые объекты, указываются их длины и теги, используемые при кодировании.

Таблица 39 — Объекты данных компонентов CDF* и RDF*

Объект	Длина (в октетах)	Тег
Зашифрованные данные (с индикатором защиты)	Не менее 2	$\texttt{87}_{16}$
Компонент Le	1 – 3	$\texttt{97}_{16}$
Имитовставка	8	$\texttt{8E}_{16}$

12.4.2 Защита команды

Команда cmd защищается с помощью алгоритма 8.5.4.2. В качестве заголовка I выступает слово CLA* ‖ INS ‖ P1 ‖ P2 (4 октета), а в качестве критического сообщения X — компонент CDF. Слово CLA* получается из CLA изменением одного бита: CLA* $\leftarrow$ CLA$\vee\texttt{04}_{16}$.

Алгоритм 8.5.4.2 настраивается следующим образом.

1 После зашифрования на шаге 1 по защищенному сообщению Y (может быть пустым) и компоненту Le (может быть пустым) строится слово Z. Для этого выполняется следующая последовательность шагов:

1) $Z \leftarrow \perp$;
2) если $|Y| > 0$, то $Z \leftarrow Z \parallel \mathtt{der}(87_{16}, 02_{16} \parallel Y)$;
3) если $|\mathrm{Le}| > 0$, то $Z \leftarrow Z \parallel \mathtt{der}(97_{16}, \mathrm{Le})$.

2 На шаге 2 кодовое представление $\langle\!\langle I, Y \rangle\!\rangle$ определяется как $I \parallel Z$.

3 После вычисления имитовставки T на шаге 3 строится слово CDF* $\leftarrow Z \parallel \mathtt{der}(8E_{16}, T)$. Затем по правилам, заданным в 12.1, определяется длина Lc* этого слова.

4 На шаге 4 кодовое представление $\langle\!\langle I, Y, T \rangle\!\rangle$ определяется как $I \parallel$ Lc* $\parallel$ CDF* $\parallel 00_{16}$. Это и есть защищенная команда cmd*.

Защита команды схематически представлена на рис. 1. Необязательные части сообщений взяты на рисунке в квадратные скобки.

CLA	INS	P1	P2	Lc	CDF	Le
				[длина CDF]	[сообщение X]	[длина ответа]

$\Downarrow$

CLA*	INS	P1	P2	Lc*	CDF*	Le*
заголовок I				длина CDF*	$[\mathtt{der}(87_{16}, 02_{16} \parallel Y) \,\|\|]\, [\mathtt{der}(97_{16}, \mathrm{Le}) \,\|\|]\, \mathtt{der}(8E_{16}, T)$	00_{16}

Рисунок 1 — Защита команды

12.4.3 Защита ответа

Ответ защищается с помощью алгоритма 8.5.4.2. В качестве заголовка I выступает слово SW1 $\parallel$ SW2 (2 октета), а в качестве критического сообщения X — компонент RDF.

Алгоритм 8.5.4.2 настраивается следующим образом.

1 После зашифрования на шаге 1 по защищенному сообщению Y (возможно пустому) строится слово Z. Для этого выполняется следующая последовательность шагов:

1) $Z \leftarrow \perp$;
2) если $|Y| > 0$, то $Z \leftarrow Z \parallel \mathtt{der}(87_{16}, 02_{16} \parallel Y)$.

2 На шаге 2 кодовое представление $\langle\!\langle I, Y \rangle\!\rangle$ определяется как $Z \parallel I$.

3 После вычисления имитовставки T на шаге 3 строится слово RDF* $\leftarrow Z \parallel \mathtt{der}(8E_{16}, T)$.

4 На шаге 4 кодовое представление $\langle\!\langle I, Y, T \rangle\!\rangle$ определяется как RDF* $\parallel I$. Это и есть защищенный ответ res*.

Защита ответа схематически представлена на рис. 2.

RDF	SW1	SW2
[сообщение X]	заголовок I	

$\Downarrow$

RDF*	SW1	SW2
$[\mathtt{der}(87_{16}, 02_{16} \parallel Y) \,\|\|]\, \mathtt{der}(8E_{16}, T)$	заголовок I	

Рисунок 2 — Защита ответа

КТ должен защищать ответ в рамках того защищенного соединения, по которому поступила защищенная команда (даже если это команда переключения между соединениями).

12.4.4 Снятие защиты

Снятие защиты с команды и ответа производится с помощью алгоритма 8.5.4.3. При снятии защиты выполняются обратные к установке защиты действия: защищенные команда cmd* и ответ res* преобразуются в исходные команду cmd и ответ res.

При снятии защиты должен проверяться формат сообщений. В частности, должно быть проверено, что в компоненте CLA* защищенной команды установлен бит защиты: $\mathrm{CLA}^* \wedge \mathtt{04}_{16} = \mathtt{04}_{16}$.

12.4.5 Принудительное закрытие защищенного соединения

КТ должен принудительно закрыть текущее защищенное соединение, если при обработке команды обнаружено, что:

1) команда передается в открытом виде;
2) отсутствует необходимый объект данных;
3) объект данных является некорректным.

В первом и втором случаях КТ должен вернуть статус $\mathrm{SW1} \parallel \mathrm{SW2} = \mathtt{6987}_{16}$, а в третьем случае — статус $\mathrm{SW1} \parallel \mathrm{SW2} = \mathtt{6988}_{16}$.

При принудительном закрытии защищенного соединения КТ должен уничтожить ключи, используемые для защиты.

Приложение А
(рекомендуемое)
Модуль АСН.1

А.1 Идентификаторы

Идентификаторы АСН.1 назначаются следующим объектам, введенным в настоящем стандарте:

`id-DocumentValidity`	дополнительный атрибут DocumentValidity (см. 7.2);
`id-AgeVerification`	дополнительный атрибут AgeVerification (см. 7.2);
`id-PlaceVerification`	дополнительный атрибут PlaceVerification (см. 7.2);
`id-eID`	прикладная программа eID (см. 6.7);
`id-eSign`	прикладная программа eSign (см. 6.8);
`id-eIdAccess`	права доступа к прикладной программе eID (см. 7.3);
`id-eSignAccess`	права доступа к прикладной программе eSign (см. 7.3);
`id-SignAuthExt`	расширение с правами доступа к прикладной программе eSign (см. 9.1);
`btok-bauth`	полный протокол BAUTH (см. 8.4);
`btok-bauth1`	протокол BAUTH с односторонней аутентификацией (см. 8.4).

Долговременному открытому ключу, который используется в протоколе BAUTH, присваивается идентификатор `bign-pubkey`, определенный в СТБ 34.101.45 (приложение Д).

А.2 Модуль

```
Btok-module-v1 {iso(1) member-body(2) by(112) 0 2 0 34 101 79 module(1) ver1(1)}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN
  IMPORTS
    PublicKey
      FROM Bign-module-v2 {iso(1) member-body(2) by(112) 0 2 0 34 101 45
        module(1) ver2(2)};

  btok OBJECT IDENTIFIER ::= {iso(1) member-body(2) by(112) 0 2 0 34 101 79}

  btok-bauth OBJECT IDENTIFIER ::= {btok 11}
  btok-bauth1 OBJECT IDENTIFIER ::= {btok 12}

  btok-attrs OBJECT IDENTIFIER ::= {btok 5}
  id-DocumentValidity OBJECT IDENTIFIER ::= {btok-attrs 1}
  id-AgeVerification OBJECT IDENTIFIER ::= {btok-attrs 2}
  id-PlaceVerification OBJECT IDENTIFIER ::= {btok-attrs 3}

  btok-access OBJECT IDENTIFIER ::= {btok 6}
  id-eIdAccess OBJECT IDENTIFIER ::= {btok-access 1}
  id-eSignAccess OBJECT IDENTIFIER ::= {btok-access 2}
```

```
btok-apps OBJECT IDENTIFIER ::= {btok 7}
id-eID OBJECT IDENTIFIER ::= {btok-apps 1}
id-eSign OBJECT IDENTIFIER ::= {btok-apps 2}

btok-cvext OBJECT IDENTIFIER ::= {btok 8}
id-SignAuthExt OBJECT IDENTIFIER ::= {btok-cvext 1}

SerialNumber ::= [APPLICATION 1] PrintableString(SIZE(14..18))
IssuingState ::= [APPLICATION 2] Country
DateOfExpiry ::= [APPLICATION 3] Date
GivenName ::= [APPLICATION 4] UTF8String
FamilyName ::= [APPLICATION 5] UTF8String
MiddleName ::= [APPLICATION 6] UTF8String
PersonalNumber ::= [APPLICATION 7] PrintableString(SIZE(7..64))
DateOfBirth ::= [APPLICATION 8] Date
PlaceOfBirth ::= [APPLICATION 9] GeneralPlace
Nationality ::= [APPLICATION 10] Country
Sex ::= [APPLICATION 11] ICAOSex
OptionalDataR ::= [APPLICATION 12] SET OF OptionalData
WrittenSignature ::= [APPLICATION 14] OCTET STRING
DateOfIssuance ::= [APPLICATION 15] Date
IssuanceBoard ::= [APPLICATION 16] UTF8String
PlaceOfResidence ::= [APPLICATION 17] GeneralPlace
DistrictID ::= [APPLICATION 18] PrintableString(SIZE(0..64))
PhoneNumber ::= [APPLICATION 21] PrintableString
EmailAddress ::= [APPLICATION 22] IA5String

ICAOString ::= PrintableString(FROM("A".."Z" | " "))
Country ::= ICAOString(SIZE(3))
ICAOSex ::= PrintableString(FROM("M"|"F"|" "))
Date ::= NumericString(SIZE(8))

Place ::= SEQUENCE {
  street  [10] UTF8String OPTIONAL,
  city    [11] UTF8String,
  state   [12] UTF8String OPTIONAL,
  country [13] Country,
  zipcode [14] PrintableString OPTIONAL
}

GeneralPlace ::= CHOICE {
  structuredPlace Place,
  freetextPlace   [1] UTF8String,
  noPlaceInfo     [2] UTF8String
}
```

```
  OptionalData ::= SEQUENCE {
    type  OBJECT IDENTIFIER,
    data  ANY DEFINED BY type OPTIONAL
  }

  AuthAuxData ::= [APPLICATION 7] SEQUENCE OF DiscretionaryDataTemplate

  DiscretionaryDataTemplate ::= [APPLICATION 19] SEQUENCE {
    objIdentifier  OBJECT IDENTIFIER,
    dataObjects    ANY DEFINED BY objIdentifier
  }

  CertHAT ::= [APPLICATION 76] SEQUENCE {
    objId              OBJECT IDENTIFIER,
    discretionaryData  OCTET STRING
  }

  CVCertificate ::= [APPLICATION 33] SEQUENCE {
    certificateBody                    [APPLICATION 78] SEQUENCE {
      certProfileIdentifier            [APPLICATION 41] INTEGER {v1(0)},
      certAuthorityReference           [APPLICATION 2]  CharString,
      publicKey                        [APPLICATION 73] PubKey,
      certHolderReference              [APPLICATION 32] CharString,
      certHolderAuthorizationTemplate [APPLICATION 76] CertHAT OPTIONAL,
      certEffectiveDate                [APPLICATION 37] CVDate,
      certExpirationDate               [APPLICATION 36] CVDate,
      certExtensions                   [APPLICATION 5]  CVExt OPTIONAL
    },
    signature                          [APPLICATION 55] OCTET STRING
  }

  CharString ::= PrintableString(SIZE(8..12))

  PubKey ::= SEQUENCE {
    objIdentifier        OBJECT IDENTIFIER,
    pubKeyandParameters  PublicKey
  }

  CVDate ::= OCTET STRING(SIZE(6)) -- YYMMDD
  CVExt ::= SEQUENCE OF DiscretionaryDataTemplate
END
```

Приложение Б
(обязательное)
Файловая система

Б.1 Общие сведения

В соответствии с [5] прикладные программы и данные КТ должны интерпретироваться как файлы следующих типов:

1 DF (dedicated file) — назначенный файл. Предназначен для объявления прикладных программ и/или объединения файлов в группы. DF может являться родительским для других файлов.

2 EF (elementary file) — элементарный файл. Предназначен для хранения данных. EF не может быть родительским для других файлов.

Имеются две категории элементарных файлов:

– внутренние, используемые КТ для целей управления и контроля;

– рабочие, предназначенные для хранения данных, к которым можно обращаться извне.

При выполнении команд КТ может обращаться к DF и EF как явно, так и неявно (автоматически). Для явного обращения используется один из следующих методов:

1 Выбор по имени. Используется только для DF. Имя представляет собой строку октетов, длина которой не превышает 16. Прикладным программам КТ назначаются специальные идентификаторы (AID). Такой идентификатор может использоваться как имя DF прикладной программы.

2 Выбор по идентификатору файла. Идентификатор файла (FID) состоит из двух октетов. Некоторые идентификаторы зарезервированы. Например, идентификатор $\texttt{3F00}_{16}$ назначается обязательному DF, который является корневым в иерархической структуре файлов и который обозначается MF (master file/мастер-файл).

Б.2 Прикладная программа eID

Прикладной программе eID назначается AID $\texttt{D11200020022654F0701}_{16}$. Он получен в результате корректировки префикса в кодовом представлении идентификатора `id-eID`.

Прикладная программа eID поддерживает 22 группы данных (DG1–DG22) с атрибутами владельца КТ (см. 7.1). Группы данных размещаются в EF, описанных в таблице Б.1. В этой и следующей таблицах R обозначает право доступа на чтение, W — на запись, R/W — на чтение и запись.

Файлы для групп данных DG1–DG11, DG15, DG16 являются обязательными: они должны формироваться при создании файловой системы КТ. Остальные файлы являются необязательными.

Таблица Б.1 — Группы данных eID

Группа	Содержание	FID	Доступ
DG1	Серийный номер КТ	$\texttt{0101}_{16}$	R
DG2	Страна, выпустившая КТ	$\texttt{0102}_{16}$	R
DG3	Дата окончания действия КТ	$\texttt{0103}_{16}$	R
DG4	Имя	$\texttt{0104}_{16}$	R
DG5	Фамилия	$\texttt{0105}_{16}$	R
DG6	Отчество	$\texttt{0106}_{16}$	R
DG7	Идентификационный номер	$\texttt{0107}_{16}$	R
DG8	Дата рождения	$\texttt{0108}_{16}$	R
DG9	Место рождения	$\texttt{0109}_{16}$	R
DG10	Гражданство	$\texttt{010A}_{16}$	R
DG11	Пол	$\texttt{010B}_{16}$	R
DG12	Дополнительные данные для чтения	$\texttt{010C}_{16}$	R
DG13	—	$\texttt{010D}_{16}$	R
DG14	Изображение ручной подписи	$\texttt{010E}_{16}$	R
DG15	Дата выпуска КТ	$\texttt{010F}_{16}$	R
DG16	Орган, выдавший КТ	$\texttt{0110}_{16}$	R
DG17	Адрес постоянного места жительства	$\texttt{0111}_{16}$	R/W
DG18	Код региона	$\texttt{0112}_{16}$	R/W
DG19	—	$\texttt{0113}_{16}$	R/W
DG20	—	$\texttt{0114}_{16}$	R/W
DG21	Номер телефона	$\texttt{0115}_{16}$	R/W
DG22	Адрес электронной почты	$\texttt{0116}_{16}$	R/W

Б.3 Прикладная программа eSign

Прикладной программе eSign назначается AID $\texttt{D11200020022654F0702}_{16}$. Он получен в результате корректировки префикса в кодовом представлении идентификатора `id-eSign`.

Прикладная программа eSign поддерживает чтение и обновление элементарных файлов, описанных в таблице Б.2.

С каждым из 6 личных ключей с идентификаторами $\texttt{01}_{16}$–$\texttt{03}_{16}$, $\texttt{11}_{16}$–$\texttt{13}_{16}$ (см. таблицу 20) связаны два элементарных файла. В первом хранится сертификат соответствующего открытого ключа (см. 6.5), во втором — хэш-значение открытого ключа. Хэш-значение рассчитывается по правилам, заданным в СТБ 34.101.78 относительно расширения `SubjectKeyIdentifier` сертификата. Хэш-значение состоит из 20 или 32 октетов.

Таблица Б.2 — Элементарные файлы eSign

Содержание	FID	Доступ	
		PS	AS
Объект `Name` (см. 12.3.5)	$\texttt{5101}_{16}$	R	R/W
Сертификат открытого ключа с идентификатором $\texttt{0X}_{16}$	$\texttt{520X}_{16}$	R/W	–
Сертификат открытого ключа с идентификатором $\texttt{1X}_{16}$	$\texttt{521X}_{16}$	–	R/W
Хэш-значение открытого ключа с идентификатором $\texttt{0X}_{16}$	$\texttt{530X}_{16}$	R/W	–
Хэш-значение открытого ключа с идентификатором $\texttt{1X}_{16}$	$\texttt{531X}_{16}$	–	R/W

Файл с сертификатом записывается извне КП или терминалом. Вместо сертификата может временно храниться хэш-значение запроса на его получение (см. 12.3.5). Файл с

хэш-значением открытого ключа либо записывается извне, либо формируется автоматически во время генерации пары ключей (см. 12.2.9).

Хэш-значение H (открытого ключа или запроса на получение сертификата) должно храниться в виде строки $\mathtt{der}(\mathtt{04}_{16}, H)$.

Файлы для объекта `Name` и ключей с идентификаторами $\mathtt{01}_{16}$, $\mathtt{11}_{16}$ являются обязательными: они должны формироваться при создании файловой системы КТ (но могут быть первоначально пустыми). Остальные файлы являются необязательными.

Приложение В
(обязательное)
Статусы ответа

В таблице В.1 приводятся типичные для команд, определенных в 12.2, ошибочные условия и соответствующие им статусы SW1 ‖ SW2, возвращаемые в ответе на команду. Правила таблицы соответствуют [5]. Специфические ошибочные условия и соответствующие статусы приводятся при описании команд в 12.2.

КТ может возвращать дополнительные статусы, не определенные в настоящем стандарте.

Таблица В.1 — Статусы ответа

Условие	SW1 ‖ SW2
Внутренняя ошибка выполнения команды	$\texttt{6400}_{16}$
Ошибка памяти при записи или чтении данных	$\texttt{6581}_{16}$
Недопустимое значение компонента Lc	$\texttt{6700}_{16}$
Значение компонента Lc не соответствует длине данных команды	$\texttt{6700}_{16}$
Компонент Lc или Le не включен в команду, когда это требуется	$\texttt{6700}_{16}$
Компонент Lc или Le включен в команду, когда это не требуется	$\texttt{6700}_{16}$
Цепочка команд не поддерживается	$\texttt{6884}_{16}$
Для текущего состояния КТ команда не может быть выполнена, или терминал не имеет права на выполнение команды	$\texttt{6982}_{16}$
Отсутствует ожидаемый объект данных защищенной команды	$\texttt{6987}_{16}$
Некорректный объект данных защищенной команды	$\texttt{6988}_{16}$
Некорректные данные в CDF	$\texttt{6A80}_{16}$
Некорректные параметры P1, P2	$\texttt{6A86}_{16}$
Ссылочные данные не найдены	$\texttt{6A88}_{16}$
Заданное значение INS не поддерживается	$\texttt{6D00}_{16}$
Заданное значение CLA не поддерживается	$\texttt{6E00}_{16}$

Библиография

[1] ISO 3166-1:1997 Code for the Representation of Names of Countries and Their Subdivisions — Part 1: Country Codes
(Коды представления названий стран и их частей. Часть 1. Коды стран)

[2] ISO/IEC 15444-1:2009 Information technology — JPEG 2000 image coding system: Core coding system, 2009
(Информационные технологии. Система кодирования изображения JPEG 2000. Ядро системы кодирования)

[3] Schulzrinne H. The tel URI for Telephone Numbers. Request for Comments: 3966, 2004
(Схема tel URI для телефонных номеров)

[4] ISO/IEC 7816-8:2005 Identification cards — Integrated circuit cards — Part 8: Commands for security operations
(Идентификационные карты. Смарт-карты. Часть 8. Команды для операций защиты)

[5] ISO/IEC 7816-4:2005 Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange
(Идентификационные карты. Смарт-карты. Часть 4. Организация, безопасность и команды обмена)

Поправка к официальной редакции

В каком месте	Напечатано	Должно быть
Пункт 8.4.6, шаг 5.4	проверяет, что $s \in \{0, 1, \ldots, q-1\}$;	проверяет, что $s_{\text{КТ}} \in \{0, 1, \ldots, q-1\}$;
Пункт 8.5.4.2, шаг 3	$T \leftarrow \texttt{belt-mac}(\langle\!\langle I, Y \rangle\!\rangle, K_1)$.	$T \leftarrow \texttt{belt-mac}(S \parallel \langle\!\langle I, Y \rangle\!\rangle, K_1)$.
Пункт 8.5.4.3, шаг 3	Проверить, что $T = \texttt{belt-mac}(\langle\!\langle I, Y, T \rangle\!\rangle, K_1)$.	Проверить, что $T = \texttt{belt-mac}(S \parallel \langle\!\langle I, Y \rangle\!\rangle, K_1)$.
Подраздел 10.3, таблица 6, шаг 3, сообщение	$\langle\!\langle \text{M0}_{\text{BPACE}}, \text{M1}_{\text{BPACE}} \rangle\!\rangle$	$\langle\!\langle \text{M1}_{\text{BPACE}} \rangle\!\rangle$
Подраздел 10.3, таблица 6, шаг 3, примечание	$\text{M0}_{\text{BPACE}} = \texttt{Chart}_{\text{B}}$	$\texttt{hello}_{\text{КП}} = \langle\!\langle \texttt{Chart}_{\text{B}} \rangle\!\rangle$
Подраздел 10.3, таблица 6, шаг 5, примечание	$\text{M0}_{\text{BAUTH}} = (H_{\text{КП}}, \texttt{Cert}(Id_{\text{T}}, Q_{\text{T}}))$	$\text{M0}_{\text{BAUTH}} = \langle\!\langle \texttt{hello}_{\text{T}}, \texttt{Cert}(Id_{\text{T}}, Q_{\text{T}}) \rangle\!\rangle$
Подраздел 10.4, шаг 3	В качестве приветственного сообщения $\texttt{hello}_{\text{КП}}$ протокола BPACE используется перечень $\texttt{Chart}_{\text{КП}}$. КТ должен сохранить $\texttt{Chart}_{\text{КП}}$ для дальнейшего использования.	В качестве приветственного сообщения $\texttt{hello}_{\text{КП}}$ протокола BPACE используется перечень $\texttt{Chart}_{\text{B}}$ и возможно другие данные. КТ должен сохранить $\texttt{Chart}_{\text{B}}$ для дальнейшего использования.
Подраздел 10.4, шаг 5	В качестве $\texttt{hello}_{\text{T}}$ используется хэш-значение $H_{\text{СИ}}$.	В $\texttt{hello}_{\text{T}}$ указывается хэш-значение $H_{\text{СИ}}$ ($H_{\text{КП}}$ на стороне КП), перечень $\texttt{Chart}_{\text{B}}$ и возможно другие данные.
Подраздел 12.1, дополнительный абзац после перечисления		В компонентах Lc и Le первый октет представляет старший байт кодируемого числа, последний октет — младший байт. Другими словами, вместо обычного для настоящего стандарта соглашения «от младших к старшим» (см. п. 4.2.2) используется соглашение «от старших к младшим» (big-endian).
Пункт 12.2.15, абзац 2	В компоненте CDF все объекты данных являются обязательными, они должны передаваться с помощью одной команды, т. е. использование цепочки команд <MSE: Set AT> не допускается. При этом порядок следования объектов данных в компоненте CDF не важен.	В компоненте CDF обязательным является только первый объект. Объекты должны передаваться с помощью одной команды, т. е. использование цепочки команд <MSE: Set AT> не допускается.

Пункт 12.2.15, абзац 3	Команда может вызываться при выборе мастер-файла в состоянии PS непосредственно после выполнения протокола BPACE (см. 12.2.7).	Команда может вызываться при выборе мастер-файла в состоянии PS непосредственно после выполнения протокола BPACE (см. 12.2.7). Приветственное сообщение $\texttt{hello}_{\text{T}}$ протокола BAUTH определяется как CDF ‖ $\texttt{CertHAT}^*$. Здесь $\texttt{CertHAT}^*$ — список объектов `CertHAT`, указанный ранее в команде инициализации протокола BPACE (см. 12.2.16). Приветственное сообщение $\texttt{hello}_{\text{KT}}$ протокола BAUTH полагается пустым.
Пункт 12.2.16	Объекты типа `CertHAT`, передаваемые в компоненте CDF, должны использоваться в протоколе BPACE как приветственное сообщение КП (см. 8.3). Если передается несколько таких значений, то при формировании приветственного сообщения они должны объединяться в порядке следования в компоненте CDF. Если же не передается ни одного значения, то в качестве приветственного сообщения должно использоваться пустое слово.	Компонент CDF должен использоваться в протоколе BPACE в качестве приветственного сообщения $\texttt{hello}_{\text{КП}}$ (см. 8.3).
Пункт 12.4.2, шаг 4	I ‖ Le* ‖ CDF* ‖ $\texttt{00}_{16}$	I ‖ Lc* ‖ CDF* ‖ $\texttt{00}_{16}$