ГОСУДАРСТВЕННЫЙ СТАНДАРТ
РЕСПУБЛИКИ БЕЛАРУСЬ

СТБ/ПР_1/34.101.45

Информационные технологии и безопасность

АЛГОРИТМЫ ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ И ТРАНСПОРТА КЛЮЧА НА ОСНОВЕ ЭЛЛИПТИЧЕСКИХ КРИВЫХ

Інфармацыйныя тэхналогіі і бяспека

АЛГАРЫТМЫ ЭЛЕКТРОННАГА ЛІЧБАВАГА ПОДПІСУ І ТРАНСПАРТУ КЛЮЧА НА АСНОВЕ ЭЛІПТЫЧНЫХ КРЫВЫХ

Госстандарт
Минск

УДК МКС 35.240.40 КП 05

Ключевые слова: электронная цифровая подпись, транспорт ключа, криптографические алгоритмы на основе эллиптических кривых

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 №

3 ВЗАМЕН СТБ П 34.101.45-2011

Издан на русском языке

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РЕСПУБЛИКИ БЕЛАРУСЬ

Информационные технологии и безопасность

АЛГОРИТМЫ ЭЛЕКТРОННОЙ ЦИФРОВОЙ ПОДПИСИ И ТРАНСПОРТА КЛЮЧА НА ОСНОВЕ ЭЛЛИПТИЧЕСКИХ КРИВЫХ

Інфармацыйныя тэхналогіі і бяспека

АЛГАРЫТМЫ ЭЛЕКТРОННАГА ЛІЧБАВАГА ПОДПІСУ І ТРАНСПОРТУ КЛЮЧА НА АСНОВЕ ЭЛІПТЫЧНЫХ КРЫВЫХ

Information technology and security

Digital signature and key transport algorithms based on elliptic curves

Дата введения 2013-XX-XX

1 Область применения

Настоящий государственный стандарт (далее — стандарт) устанавливает алгоритмы выработки и проверки электронной цифровой подписи (далее — ЭЦП), алгоритмы транспорта ключа, а также сопровождающие их алгоритмы генерации и проверки параметров эллиптической кривой, генерации личных и открытых ключей подписи, проверки открытых ключей.

Настоящий стандарт применяется при разработке средств криптографической защиты информации, в том числе средств ЭЦП и шифрования.

2 Нормативные ссылки

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

СТБ 1176.2-99 Информационная технология. Защита информации. Процедуры выработки и проверки электронной цифровой подписи

СТБ 34.101.17-2012 Информационные технологии и безопасность. Синтаксис запроса на получение сертификата

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

СТБ 34.101.23-2012 Информационные технологии и безопасность. Синтаксис криптографических сообщений

СТБ 34.101.26-2012 Информационные технологии и безопасность. Онлайновый протокол проверки статуса сертификата

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

СТБ 34.101.47-2012 Информационные технологии и безопасность. Криптографические алгоритмы генерации псевдослучайных чисел

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

Проект, предварительная редакция

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 замененными (измененными) 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 следующие термины с соответствующими определениями:

3.1 ключ: Параметр, который управляет криптографическими операциями зашифрования и расшифрования, выработки и проверки ЭЦП, генерации псевдослучайных чисел и др.

3.2 личный ключ: Ключ, который связан с конкретной стороной, не является общедоступным и используется в настоящем стандарте для выработки ЭЦП и для разбора токена ключа.

3.3 октет: Двоичное слово длины 8.

3.4 открытый ключ: Ключ, который строится по личному ключу, связан с конкретной стороной, может быть сделан общедоступным и используется в настоящем стандарте для проверки ЭЦП и для создания токена ключа.

3.5 синхропосылка: Открытые входные данные криптографического алгоритма, которые обеспечивают уникальность результатов криптографического преобразования на фиксированном ключе.

3.6 сообщение: Двоичное слово конечной длины.

3.7 токен ключа: Сообщение, которое передается от одной стороны другой при транспорте ключа.

3.8 транспорт ключа: Конфиденциальная передача ключа от одной стороны другой.

3.9 хэш-значение: Двоичное слово фиксированной длины, которое определяется по сообщению без использования ключа и служит для контроля целостности сообщения и для представления сообщения в сжатой форме.

3.10 хэширование: Выработка хэш-значений.

3.11 электронная цифровая подпись; ЭЦП: Двоичное слово, которое служит для контроля целостности и подлинности сообщения, обеспечивает невозможность отказа от

авторства, определяется с использованием личного ключа и проверяется с использованием открытого ключа.

4 Обозначения

4.1 Список обозначений

$\{0,1\}^n$	множество всех слов длины n в алфавите $\{0,1\}$;
$\{0,1\}^*$	множество всех слов конечной длины в алфавите $\{0,1\}$ (включая пустое слово 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 $v = v_1v_2\dots v_m$;
$\mathtt{01234...}_{16}$	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 $x-r$;
$x \equiv y \pmod m$	x сравнимо с y по модулю m, т. е. $x \bmod m = y \bmod m$;
$u \oplus v$	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 $w_i = (u_i + v_i) \bmod 2$;
$\bar{u}$	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 $\bar{u}_1 + 2^8\bar{u}_2 + \dots + 2^{8(n-1)}\bar{u}_n$;
$\langle U\rangle_{8n}$	для целого числа U слово $u \in \{0,1\}^{8n}$ такое, что $\bar{u} = U \bmod 2^{8n}$;
$u \boxplus v$	для $u, v \in \{0,1\}^{8n}$ слово $\langle \bar{u} + \bar{v}\rangle_{8n}$;
$\mathbb{F}_p$	для простого числа p множество $\{0,1,\dots,p-1\}$ с операциями сложения и умножения по модулю p, конечное поле из p элементов;
$\left(\frac{u}{p}\right)$	для нечетного простого числа p и $u \in \mathbb{F}_p$ символ Лежандра: 0, если $u = 0$; 1, если u — квадратичный вычет по модулю p, и -1 в остальных случаях;
$E^*_{a,b}(\mathbb{F}_p)$	для $a, b \in \mathbb{F}_p$ множество решений (x, y), $x, y \in \mathbb{F}_p$, уравнения $y^2 = x^3 + ax + b$, множество аффинных точек эллиптической кривой;
O	бесконечно удаленная точка;
$E_{a,b}(\mathbb{F}_p)$	множество $E^*_{a,b}(\mathbb{F}_p) \cup \{O\}$ с операцией сложения точек, группа точек эллиптической кривой;
kP	для $P \in E_{a,b}(\mathbb{F}_p)$ сумма k экземпляров P, кратная P точка;
l	уровень стойкости, число из множества $\{128, 192, 256\}$;

$\langle P \rangle$	для $P = (x, y) \in E^*_{a,b}(\mathbb{F}_p)$, где $2^{2l-1} < p < 2^{2l}$, слово $\langle x \rangle_{2l} \parallel \langle y \rangle_{2l}$;
$\langle P \rangle_n$	для $P \in E^*_{a,b}(\mathbb{F}_p)$ слово из первых n символов $\langle P \rangle$, $n \leqslant \lvert\langle P \rangle\rvert$;
$c \leftarrow u$	присвоение переменной c значения u;
$c \xleftarrow{R} U$	случайный равновероятный (или псевдослучайный) выбор c из множества U;
`belt-hash`	алгоритм хэширования, определенный в СТБ 34.101.31 (пункт 6.9.3);
$\mathrm{OID}(D)$	кодовое представление идентификатора объекта D, полученное в соответствии с ГОСТ 34.973, ГОСТ 34.974.

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

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

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

4.2.2 Слова как числа

Октету $u = u_1 u_2 \dots u_8$ ставится в соответствие байт — число $\bar{u} = 2^7 u_1 + 2^6 u_2 + \dots + u_8$. 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 является младшим, последний — старшим. Например, слову w соответствует число

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

4.2.3 Конечные поля

Элементы $\mathbb{F}_p$ складываются и умножаются как целые числа с заменой результата на остаток от его деления на p. Множество $\mathbb{F}_p$ с такими операциями является конечным простым полем. Нулевым элементом поля является число 0, а мультипликативной единицей — число 1 (подробнее см. [1]).

Кроме сложения и умножения, в поле $\mathbb{F}_p$ можно выполнять вычитание и деление. Вычитание u состоит в сложении с $p-u$. Деление на $u \in \{1, 2, \ldots, p-1\}$ состоит в умножении на число $v \in \{1, 2, \ldots, p-1\}$ такое, что $uv \equiv 1 \pmod{p}$.

Например, в поле $\mathbb{F}_7$ выполняется:

$$4 + 5 = 2, \quad 4 \cdot 5 = 6, \quad 4 - 5 = 4 + (7 - 5) = 6, \quad 4/5 = 4 \cdot 3 = 5.$$

Квадраты ненулевых элементов $\mathbb{F}_p$ называются квадратичными вычетами по модулю p. Например, имеется 3 квадратичных вычета по модулю 7:

$$1 = 1^2, \quad 2 = 3^2, \quad 4 = 2^2.$$

Поэтому $\left(\frac{1}{7}\right) = \left(\frac{2}{7}\right) = \left(\frac{4}{7}\right) = 1$. Кроме того, $\left(\frac{0}{7}\right) = 0$ и $\left(\frac{3}{7}\right) = \left(\frac{5}{7}\right) = \left(\frac{6}{7}\right) = -1$.

4.2.4 Эллиптические кривые

Пусть $p > 3$ и $4a^3 + 27b^2 \not\equiv 0 \pmod{p}$. Множество $E^*_{a,b}(\mathbb{F}_p)$ состоит из решений уравнения $y^2 = x^3 + ax + b$ относительно $x, y \in \mathbb{F}_p$. Уравнение такого вида определяет эллиптическую кривую над полем $\mathbb{F}_p$, его решения (x, y) называются аффинными точками кривой. К аффинным точкам добавляется специальная бесконечно удаленная точка O и образуется множество $E_{a,b}(\mathbb{F}_p)$. Например,

$$E_{4,1}(\mathbb{F}_7) = \{O, (0,1), (0,6), (4,2), (4,5)\}.$$

Множество $E_{a,b}(\mathbb{F}_p)$ является аддитивной группой при следующих правилах сложения:

1 $O + P = P + O = P$ для всех $P \in E_{a,b}(\mathbb{F}_p)$.

2 Если $P = (x, y) \in E^*_{a,b}(\mathbb{F}_p)$, то $-P = (x, p - y)$ и $P + (-P) = O$.

3 Если $P_1 = (x_1, y_1) \in E^*_{a,b}(\mathbb{F}_p)$, $P_2 = (x_2, y_2) \in E^*_{a,b}(\mathbb{F}_p)$ и $P_2 \neq -P_1$, то $P_1 + P_2 = (x_3, y_3)$, где $x_3 = \lambda^2 - x_1 - x_2$, $\quad y_3 = \lambda(x_1 - x_3) - y_1$, $\quad \lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & P_1 \neq P_2, \\ \dfrac{3x_1^2 + a}{2y_1}, & P_1 = P_2 \end{cases}$

(вычисления ведутся в $\mathbb{F}_p$).

Сумма k экземпляров точки P называется k-кратной ей точкой и обозначается через kP. Например, для $P = (4, 2) \in E_{4,1}(\mathbb{F}_7)$ ее кратные имеют вид:

$$2P = (4,2) + (4,2) = (0,1), \qquad 3P = (0,1) + (4,2) = (0,6),$$
$$4P = 2(0,1) = (4,5), \qquad 5P = (0,1) + (0,6) = O.$$

Считается, что $0P = O$.

4.2.5 Идентификаторы объектов

В ГОСТ 34.973 определены правила абстрактно-синтаксической нотации версии 1 для описания различных информационных объектов. Эти правила регламентируют, в том числе присвоение объектам уникальных идентификаторов.

Идентификатор объекта представляет собой последовательность целых чисел. При записи идентификатора числа разделяются пробелами. Вся последовательность окаймляется фигурными скобками. Например, идентификатор алгоритма хэширования `belt-hash` определен в СТБ 34.101.31 как {1 2 112 0 2 0 34 101 31 81}.

Идентификатор объекта кодируется двоичным словом по правилам, заданным в ГОСТ 34.974 и кратко изложенным в приложении А. Например, OID(`belt-hash`) = $\texttt{06092A7000020022651F51}_{16}$.

5 Общие положения

5.1 Назначение

Настоящий стандарт определяет алгоритмы ЭЦП, которые предназначены для контроля целостности и подлинности сообщений. Автор сообщения использует свой личный ключ для выработки ЭЦП, а связанный с личным ключом открытый ключ используется другими сторонами для проверки ЭЦП. При правильном управлении ключами корректность проверяемой подписи означает, что она была выработана владельцем личного ключа и после этого сообщение не изменялось. Только владелец личного ключа может выработать корректную ЭЦП, что не позволяет ему отказаться от авторства сообщения и может быть использовано другими сторонами для доказательства такого авторства.

Примечание — Алгоритмы ЭЦП установлены также в СТБ 1176.2. Переход от алгоритмов СТБ 1176.2 к алгоритмам стандарта позволит уменьшить время выработки и проверки ЭЦП, сократить длины параметров и ключей при сохранении уровня криптографической стойкости.

Алгоритмы выработки и проверки ЭЦП построены по схеме Шнорра [6]. При выполнении алгоритмов используются вычисления в группе точек эллиптической кривой над конечным простым полем. В стандарте определяются алгоритмы генерации и проверки параметров, описывающих искомую группу. Определены также алгоритм генерации пары ключей (личного и открытого) и алгоритм проверки открытого ключа.

Алгоритмы проверки параметров эллиптической кривой и открытого ключа следует применять в тех случаях, когда отсутствует гарантия их математической корректности. Такая гарантия обеспечивает достоверность выводов о стойкости алгоритмов ЭЦП. Вместе с тем алгоритм проверки открытого ключа не гарантирует, что ключ действительно принадлежит определенной стороне или что сторона знает соответствующий личный ключ. Проверка знания личного ключа, удостоверение принадлежности открытого ключа и проверка такой принадлежности реализуются с помощью дополнительных методов и средств, в совокупности называемых инфраструктурой открытых ключей. Например, в СТБ 34.101.19 определяются элементы инфраструктуры на основе сертификатов открытых ключей.

Параметры эллиптической кривой, личный и открытый ключи могут быть использованы не только для контроля целостности и подлинности, но и для обеспечения конфиденциальности. В стандарте определяются алгоритмы транспорта ключа, предназначенные для защищенной передачи ключей и других секретных данных между двумя сторонами. С помощью транспортируемого ключа стороны могут выполнять шифрование или другие криптографические операции.

Для реализации транспорта отправитель вызывает алгоритм создания токена ключа. Токен представляет собой сообщение, которое включает транспортируемый ключ в защищенной форме, а также данные, необходимые получателю для снятия защиты. Получатель вызывает алгоритм разбора токена и восстанавливает транспортируемый ключ. При создании токена отправитель использует открытый ключ получателя. При разборе токена получатель использует свой личный ключ.

В приложении Б приводятся стандартные наборы параметров эллиптической кривой, которые были получены с помощью соответствующего алгоритма генерации и могут быть использованы напрямую, без повторного построения.

В приложении В определяются алгоритмы идентификационной ЭЦП, с помощью которых в некоторых случаях можно упростить управление открытыми ключами.

В приложении Г приводятся примеры выполнения алгоритмов стандарта. Примеры можно использовать для проверки корректности реализаций алгоритмов.

В приложении Д приводится модуль абстрактно-синтаксической нотации версии 1 (АСН.1), определенной в ГОСТ 34.973. Модуль задает идентификаторы алгоритмов и других объектов стандарта, описывает структуры данных для хранения ключей и параметров. Рекомендуется использовать модуль при встраивании алгоритмов стандарта в информационные системы, в которых также используется АСН.1. В частности, модуль может быть использован для уточнения форматов запроса на получение сертификата (определен в СТБ 34.101.17), сертификатов и списков отозванных сертификатов (СТБ 34.101.19), криптографических сообщений (СТБ 34.101.23), запроса и ответа о статусе сертификата (СТБ 34.101.26).

В приложении Е определяются алгоритмы защиты личного ключа на пароле его владельца. Алгоритмы соответствуют стандарту [4].

5.2 Уровень стойкости

Алгоритмы ЭЦП построены так, что злоумышленнику вычислительно трудно решить задачу подделки ЭЦП. В этой задаче злоумышленник получает параметры эллиптической кривой и открытый ключ ЭЦП. Злоумышленник не знает личный ключ, но может передавать для подписи на нем произвольные сообщения, получать и анализировать результаты. Ему требуется построить корректную ЭЦП к любому сообщению, отличному от ранее подписанных.

Стойкость алгоритмов ЭЦП определяется уровнем $l \in \{128, 192, 256\}$. На уровне l для подделки ЭЦП злоумышленнику требуется выполнить порядка 2^l операций. Стойкость основывается на сложности дискретного логарифмирования в группе точек эллиптической кривой и на стойкости используемых функций хэширования.

Уровень l определяет длины параметров, ключей, подписей и, соответственно, быстродействие алгоритмов ЭЦП. Следует учитывать, что с ростом l, кроме повышения стойкости, снижается быстродействие алгоритмов.

Для алгоритмов транспорта ключа вводятся аналогичные уровни стойкости $l \in \{128, 192, 256\}$. На уровне l для определения транспортируемого ключа по токену и открытому ключу получателя злоумышленнику требуется выполнить порядка 2^l операций.

5.3 Параметры эллиптической кривой

Модуль p**.** Используется простое число p, которое удовлетворяет условиям: $2^{2l-1} < p < 2^{2l}$, $p \equiv 3 \pmod 4$. Модуль определяет поле $\mathbb{F}_p$, над которым строится эллиптическая кривая. Можно использовать произвольное допустимое p, в том числе простое специального вида.

Коэффициенты a**,** b**.** Используются числа $a, b \in \mathbb{F}_p$, которые удовлетворяют условиям: $a \neq 0$, $\left(\frac{b}{p}\right) = 1$, $4a^3 + 27b^2 \not\equiv 0 \pmod p$. Коэффициенты a, b вместе с модулем p определяют группу точек эллиптической кривой $E_{a,b}(\mathbb{F}_p)$.

Параметр *seed***.** Числа p и a выбираются, а b строится по ним. При построении b используется дополнительный параметр $seed \in \{0,1\}^{64}$, который может быть выбран произвольным образом.

Порядок q**.** После построения группы $E_{a,b}(\mathbb{F}_p)$ рассчитывается ее порядок $q = |E_{a,b}(\mathbb{F}_p)|$. Выбирается группа, порядок которой удовлетворяет следующим ограничениям: q — простое, $2^{2l-1} < q < 2^{2l}$, $q \neq p$, q не делит числа вида $p^m - 1$ для $m = 1, 2, \ldots, 50$.

Базовая точка G**.** Используется базовая точка $G \in E^*_{a,b}(\mathbb{F}_p)$ вида $G = (0, y_G)$, где $y_G = b^{(p+1)/4} \bmod p$. Кратные G, $2G$,..., $(q-1)G$ базовой точки пробегают все элементы $E^*_{a,b}(\mathbb{F}_p)$, а $qG = O$.

Алгоритм генерации параметров эллиптической кривой определен в 6.1.3. Алгоритм проверки параметров определен в 6.1.4.

Примечание — Алгоритм генерации параметров имеет высокую вычислительную сложность. Основные издержки связаны с расчетом порядка q. Алгоритм проверки параметров имеет значительно меньшую сложность, поскольку требуется проверять, а не определять q.

5.4 Ключи

Личным ключом является число $d \in \{1, 2, \ldots, q-1\}$. По личному ключу определяется открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$. Алгоритм генерации личного и открытого ключей определен в 6.2.2. Алгоритм проверки открытого ключа определен в 6.2.3.

При хранении и распространении должны обеспечиваться конфиденциальность и контроль целостности личного ключа, контроль целостности открытого ключа. Применяемые методы управления ключами должны гарантировать принадлежность открытого ключа стороне, подпись которой проверяется, и знание данной стороной соответствующего личного ключа.

Кроме личного ключа d, в алгоритме выработки ЭЦП используется одноразовый личный ключ $k \in \{1, 2, \ldots, q-1\}$. Одноразовый ключ используется также в алгоритме создания токена ключа. Одноразовые личные ключи должны вырабатываться без возможности предсказания и уничтожаться после использования.

Для создания личных ключей может быть использован физический генератор случайных чисел, удовлетворяющий ТНПА, или алгоритм генерации псевдослучайных чисел, определенный в СТБ 34.101.47 или в другом ТНПА. Входные данные алгоритма должны включать секретный ключ, известный только владельцу личного ключа, и уникальную синхропосылку. Длина ключа алгоритма генерации должна быть не меньше l.

Примечание — Личные ключи — числа из множества $\{1, 2, \dots, q-1\}$ — генерируются, как правило, в два этапа: сначала строятся случайные или псевдослучайные двоичные слова, которые затем преобразуются в числа. Для генерации личных ключей по такой схеме рекомендуется строить слова $u \in \{0,1\}^{2l}$ до тех пор, пока не будет выполнено условие $\bar{u} \in \{1, 2, \dots, q-1\}$, и объявлять окончательное число $\bar{u}$ результатом генерации. В среднем потребуется проверить $2^{2l}/(q-1)$ чисел-кандидатов.

Для создания одноразовых личных ключей при выработке ЭЦП может использоваться алгоритм генерации, определенный в 6.3.3. Ключом данного алгоритма является d, а синхропосылкой — хэш-значение подписываемого сообщения.

Один и тот же ключ d разрешено использовать как в алгоритме выработки ЭЦП (в том числе для генерации одноразового ключа), так и в алгоритме разбора токена ключа. Использование d в других алгоритмах запрещено.

В информационных системах ключи представляются двоичными словами. Для обеспечения совместимости рекомендуется представлять личный ключ d словом $\langle d \rangle_{2l}$, а открытый ключ Q — словом $\langle Q \rangle_{4l}$.

5.5 Функция хэширования

В алгоритмах выработки и проверки ЭЦП используется функция хэширования h, которая ставит в соответствие подписываемому или проверяемому сообщению X его хэш-значение $h(X)$.

На уровне стойкости l должна использоваться функция h, значениями которой являются двоичные слова длины $2l$. Например, при $l = 128$ в качестве h можно выбрать функцию, заданную алгоритмом `belt-hash`.

Функция h должна быть алгоритмически определена в некотором ТНПА. Алгоритму хэширования в ТНПА должен быть назначен уникальный идентификатор. Кодовое представление $\mathrm{OID}(h)$ этого идентификатора используется в алгоритмах ЭЦП.

Идентификатор функции хэширования должен полностью определять ее действие. Недопустимы ситуации, когда для описания действия кроме идентификатора требуется указывать дополнительные параметры, например, начальное хэш-значение.

5.6 Транспорт ключа

В алгоритмах транспорта ключа используется заголовок ключа. Заголовок представляет собой слово $I \in \{0,1\}^{128}$, которое описывает открытые атрибуты транспортируемого ключа, включая данные об отправителе или получателе. Заголовок может передаваться вместе с токеном ключа. Если необходимости в передаче атрибутов ключей нет, то могут использоваться постоянные заголовки, которые не требуется передавать. По умолчанию $I = 0^{128}$.

Один и тот же ключ может транспортироваться одновременно нескольким сторонам. В этом случае отправитель должен создать токены ключа для каждой из сторон. Если стороны-получатели используют одинаковые параметры эллиптической кривой, то при создании токенов может использоваться один и тот же одноразовый личный ключ k.

6 Алгоритмы управления параметрами и ключами

6.1 Генерация и проверка параметров эллиптической кривой

6.1.1 Входные и выходные данные

Входными данными алгоритма генерации параметров эллиптической кривой являются уровень стойкости $l \in \{128, 192, 256\}$, простой модуль p и целый коэффициент a. Должны выполняться следующие условия: $2^{2l-1} < p < 2^{2l}$, $p \equiv 3 \pmod 4$, $0 < a < p$.

Выходными данными алгоритма генерации параметров являются параметр $seed \in \{0,1\}^{64}$, коэффициент b $(0 < b < p)$, порядок q $(2^{2l-1} < q < 2^{2l})$ и базовая точка $G \in E^*_{a,b}(\mathbb{F}_p)$.

Входными данными алгоритма проверки параметров эллиптической кривой являются модуль p, коэффициенты a и b, параметр $seed$, порядок q и базовая точка G. Параметры p, a, b, q являются целыми числами, $seed \in \{0,1\}^{64}$, точка G задается двумя целыми координатами.

Выходными данными алгоритма проверки параметров является ответ `ДА` или `НЕТ`. Ответ `ДА` означает, что переданные параметры описывают допустимую группу точек эллиптической кривой и были сгенерированы надлежащим образом. Ответ `НЕТ` означает обратное.

6.1.2 Вспомогательные алгоритмы и переменные

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, определенный в СТБ 34.101.31 (пункт 6.9.3). Алгоритм берет на вход слово $X \in \{0,1\}^*$ и возвращает его хэш-значение $Y \in \{0,1\}^{256}$.

Вычисление порядка группы точек. На шаге 6 алгоритма генерации параметров определяется порядок группы точек эллиптической кривой. Для вычисления порядка может быть использован алгоритм Шуфа или его модернизации, например алгоритм Шуфа — Элкиса — Аткина (см. [7], пункт 4.2.3).

Проверка простоты. В перечислении 2) на шаге 7 алгоритма генерации параметров и в перечислении 3) на шаге 2 алгоритма проверки параметров контролируется простота чисел. Для проверки простоты рекомендуется использовать алгоритм Рабина — Миллера, определенный в Ж.1.3.

Тест на квадратичный вычет. В перечислении 1) на шаге 5 алгоритма генерации параметров и в перечислении 4) на шаге 5 алгоритма проверки параметров контролируется, что число b является квадратичным вычетом по модулю p. Для этого рекомендуется использовать алгоритм вычисления символа Лежандра, определенный в Ж.2.3.

Переменная t. Используется переменная $t \in \{0,1\}^{4l}$.

Переменная B. Используется переменная $B \in \{0,1\}^{512}$.

6.1.3 Алгоритм генерации параметров эллиптической кривой

Генерация параметров эллиптической кривой состоит в выполнении следующих шагов:

1 Выбрать произвольным образом $seed$.

2 Установить $t \leftarrow \langle p \rangle_{2l} \,\|\, \langle a \rangle_{2l}$.

3 Установить $B \leftarrow \texttt{belt-hash}(t \,\|\, seed) \,\|\, \texttt{belt-hash}(t \,\|\, seed \boxplus \langle 1 \rangle_{64})$.

4 Установить $b \leftarrow \overline{B} \mod p$.

5 Если нарушается одно из условий:

1) $\left(\frac{b}{p}\right) = 1$;

2) $4a^3 + 27b^2 \not\equiv 0 \pmod p$,

то вернуться к шагу 1.

6 Установить $q \leftarrow |E_{a,b}(\mathbb{F}_p)|$.

7 Если нарушается одно из условий:

1) $2^{2l-1} < q < 2^{2l}$;

2) q — простое;

3) $p \neq q$;

4) $p^m \not\equiv 1 \pmod q$ для $m = 1, 2, \ldots, 50$,

то вернуться к шагу 1.

8 Установить $G \leftarrow (0, b^{(p+1)/4} \bmod p)$.

9 Возвратить $(seed, b, q, G)$.

6.1.4 Алгоритм проверки параметров эллиптической кривой

Проверка параметров эллиптической кривой состоит в выполнении следующих шагов:

1 Определить l как минимальное натуральное число, для которого $p < 2^{2l}$.

2 Если нарушается одно из условий:

1) $l \in \{128, 192, 256\}$;

2) $2^{2l-1} < p, q < 2^{2l}$;

3) p, q — простые;

4) $p \equiv 3 \pmod 4$;

5) $q \neq p$;

6) $p^m \not\equiv 1 \pmod q$ для $m = 1, 2, \ldots, 50$,

то возвратить `НЕТ`.

3 Установить $t \leftarrow \langle p \rangle_{2l} \,\|\, \langle a \rangle_{2l}$.

4 Установить $B \leftarrow \texttt{belt-hash}(t \,\|\, seed) \,\|\, \texttt{belt-hash}(t \,\|\, seed \boxplus \langle 1 \rangle_{64})$.

5 Если нарушается одно из условий:

1) $0 < a, b < p$;

2) $b \equiv \overline{B} \pmod p$;

3) $4a^3 + 27b^2 \not\equiv 0 \pmod p$;

4) $\left(\frac{b}{p}\right) = 1$;

5) $G = (0, b^{(p+1)/4} \bmod p)$;

6) $qG = O$,

то возвратить `НЕТ`.

6 Возвратить ДА.

6.2 Генерация и проверка ключей

6.2.1 Входные и выходные данные

Входными данными алгоритма генерации пары ключей являются параметры p, a, b, q, G, которые описывают группу точек эллиптической кривой. Параметры должны удовлетворять условиям алгоритма 6.1.4.

Выходными данными алгоритма генерации пары ключей являются личный ключ $d \in \{1, 2, \ldots, q-1\}$ и соответствующий открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$.

Входными данными алгоритма проверки открытого ключа являются параметры p, a, b, которые описывают группу точек эллиптической кривой, и открытый ключ $Q = (x_Q, y_Q)$, где x_Q, y_Q — целые числа. Параметры эллиптической кривой должны удовлетворять условиям алгоритма 6.1.4.

Выходными данными алгоритма проверки открытого ключа является ответ ДА или НЕТ. Ответ ДА означает, что Q является допустимым открытым ключом. Ответ НЕТ означает обратное.

6.2.2 Алгоритм генерации пары ключей

Генерация пары ключей состоит в выполнении следующих шагов:

1 Выработать $d \xleftarrow{R} \{1, 2, \ldots, q-1\}$ (в соответствии с требованиями пункта 5.4).

2 Установить $Q \leftarrow dG$.

3 Возвратить (d, Q).

6.2.3 Алгоритм проверки открытого ключа

Проверка открытого ключа состоит в выполнении следующих шагов:

1 Если нарушается одно из условий:

1) $0 \leqslant x_Q, y_Q < p$;

2) $y_Q^2 \equiv x_Q^3 + ax_Q + b \pmod p$,

то возвратить НЕТ.

2 Возвратить ДА.

6.3 Генерация одноразового личного ключа

6.3.1 Входные и выходные данные

Входными данными алгоритма генерации одноразового личного ключа являются порядок q группы точек эллиптической кривой, $2^{2l-1} < q < 2^{2l}$, личный ключ $d \in \{1, 2, \ldots, q-1\}$ и хэш-значение $H = h(X) \in \{0,1\}^{2l}$ подписываемого сообщения X (см. шаг 1 алгоритма 7.1.3). Дополнительно используется идентификатор $\mathrm{OID}(h)$. По входным данным определяется число $n = l/64$ ($n = 2$, 3 или 4).

Выходными данными алгоритма является одноразовый личный ключ $k \in \{1, 2, \ldots, q-1\}$.

6.3.2 Вспомогательные алгоритмы и переменные

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, описанный в 6.1.2.

Алгоритм `belt-block`. Используется алгоритм зашифрования `belt-block`, определенный в СТБ 34.101.31 (шифрование блока, пункт 6.1.3). Алгоритм берет на вход слово $X \in \{0,1\}^{128}$, ключ $\theta \in \{0,1\}^{256}$ и возвращает зашифрованное слово $Y \in \{0,1\}^{128}$.

Переменная r. Используется переменная $r = r_1 \parallel r_2 \parallel \ldots \parallel r_n$, где $r_i \in \{0,1\}^{128}$. Значение r должно быть уничтожено после использования.

Переменная s. Используется переменная $s \in \{0,1\}^{128}$. Значение s должно быть уничтожено после использования.

Переменная t. Используется переменная $t \in \{0,1\}^*$. Значение t выбирается произвольным, в том числе случайным или псевдослучайным, образом. Может использоваться фиксированное значение t. По умолчанию t полагается пустым словом.

Переменная θ. Используется переменная $\theta \in \{0,1\}^{256}$. Значение θ должно быть уничтожено после использования.

6.3.3 Алгоритм генерации одноразового личного ключа

Генерация одноразового личного ключа состоит в выполнении следующих шагов:

1 Выбрать t произвольным образом.

2 Установить $\theta \leftarrow \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle d \rangle_{2l} \parallel t)$.

3 Установить $r \leftarrow H$.

4 Для $i = 1, 2, \ldots$ выполнить:

1) если $n = 2$, то
   (a) $s \leftarrow r_1$;
2) если $n = 3$, то
   (a) $s \leftarrow r_1 \oplus r_2$;
   (b) $r_1 \leftarrow r_2$;
3) если $n = 4$, то
   (a) $s \leftarrow r_1 \oplus r_2 \oplus r_3$;
   (b) $r_1 \leftarrow r_2$;
   (c) $r_2 \leftarrow r_3$;
4) $r_{n-1} \leftarrow \texttt{belt-block}(s, \theta) \oplus r_n \oplus \langle i \rangle_{128}$;
5) $r_n \leftarrow s$;
6) если i кратно $2n$ и $\overline{r} \in \{1, 2, \ldots, q-1\}$, то перейти к шагу 5.

5 Установить $k \leftarrow \overline{r}$.

6 Возвратить k.

7 Основные алгоритмы

7.1 Выработка и проверка электронной цифровой подписи

7.1.1 Входные и выходные данные

Входными данными алгоритмов ЭЦП являются параметры p, a, b, q, G, которые описывают группу точек эллиптической кривой. Параметры должны удовлетворять условиям

алгоритма 6.1.4. По модулю p определяется уровень стойкости l как минимальное натуральное число, для которого $p < 2^{2l}$.

Кроме параметров эллиптической кривой, входными данными алгоритма выработки ЭЦП являются сообщение $X \in \{0,1\}^*$ и личный ключ $d \in \{1, 2, \ldots, q-1\}$.

Выходными данными алгоритма выработки ЭЦП является слово $S \in \{0,1\}^{3l}$ — подпись X.

Кроме параметров эллиптической кривой, входными данными алгоритма проверки ЭЦП являются сообщение $X \in \{0,1\}^*$, подпись $S \in \{0,1\}^*$ и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$. Открытый ключ Q должен удовлетворять условиям алгоритма 6.2.3.

Выходными данными алгоритма проверки ЭЦП является ответ `ДА` или `НЕТ`. Ответ `ДА` означает, что S является корректной подписью X. Ответ `НЕТ` означает обратное.

7.1.2 Вспомогательные алгоритмы и преобразования, переменные

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, описанный в 6.1.2.

Функция h. Используется функция хэширования h, которая действует из $\{0,1\}^*$ в $\{0,1\}^{2l}$. Требования к h определены в 5.5.

Одноразовый личный ключ k. При выработке ЭЦП используется одноразовый личный ключ $k \in \{1, 2, \ldots, q-1\}$. Требования по управлению k определены в 5.4.

Переменная H. Используется переменная $H \in \{0,1\}^{2l}$.

Переменная R. Используется переменная $R \in E_{a,b}(\mathbb{F}_p)$.

Переменная t. При проверке ЭЦП используется переменная $t \in \{0,1\}^l$.

7.1.3 Алгоритм выработки электронной цифровой подписи

ЭЦП составляется из частей $S_0 \in \{0,1\}^l$ и $S_1 \in \{0,1\}^{2l}$. Выработка ЭЦП состоит в выполнении следующих шагов:

1 Установить $H \leftarrow h(X)$.
2 Выработать $k \xleftarrow{R} \{1, 2, \ldots, q-1\}$ (в соответствии с требованиями пункта 5.4).
3 Установить $R \leftarrow kG$.
4 Установить $S_0 \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle R \rangle_{2l} \parallel H) \big\rangle_l$.
5 Установить $S_1 \leftarrow \big\langle (k - \overline{H} - (\overline{S}_0 + 2^l)d) \bmod q \big\rangle_{2l}$.
6 Установить $S \leftarrow S_0 \parallel S_1$.
7 Возвратить S.

7.1.4 Алгоритм проверки электронной цифровой подписи

Проверка ЭЦП состоит в выполнении следующих шагов:

1 Если $|S| \neq 3l$, то возвратить `НЕТ`.
2 Представить S в виде $S = S_0 \parallel S_1$, где $S_0 \in \{0,1\}^l$, $S_1 \in \{0,1\}^{2l}$.
3 Если $\overline{S}_1 \geqslant q$, то возвратить `НЕТ`.
4 Установить $H \leftarrow h(X)$.
5 Установить $R \leftarrow \big((\overline{S}_1 + \overline{H}) \bmod q\big) G + (\overline{S}_0 + 2^l)Q$.
6 Если $R = O$, то возвратить `НЕТ`.
7 Установить $t \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle R \rangle_{2l} \parallel H) \big\rangle_l$.

8 Если $S_0 \neq t$, то возвратить `НЕТ`.

9 Возвратить `ДА`.

7.2 Транспорт ключа

7.2.1 Входные и выходные данные

Входными данными алгоритмов транспорта ключа являются параметры p, a, b, q, G, которые описывают группу точек эллиптической кривой. Параметры должны удовлетворять условиям алгоритма 6.1.4. По модулю p определяется уровень стойкости l как минимальное натуральное число, для которого $p < 2^{2l}$.

Кроме параметров эллиптической кривой, входными данными алгоритма создания токена являются транспортируемый 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. Открытый ключ Q должен удовлетворять условиям алгоритма 6.2.3.

Выходными данными алгоритма создания токена является слово $Y \in \{0,1\}^{2l+|X|+128}$ — токен ключа X.

Кроме параметров эллиптической кривой, входными данными алгоритма разбора токена являются токен $Y \in \{0,1\}^*$, заголовок $I \in \{0,1\}^{128}$ транспортируемого в нем ключа и личный ключ $d \in \{1,2,\ldots,q-1\}$ получателя токена.

Выходными данными алгоритма разбора токена является либо признак `ОШИБКА`, либо слово $X \in \{0,1\}^{|Y|-2l-128}$ — ключ, который транспортируется в токене Y. Возврат признака `ОШИБКА` означает некорректность токена.

7.2.2 Вспомогательные алгоритмы и переменные

Алгоритм `belt-keywrap`. Используется алгоритм `belt-keywrap`, определенный в СТБ 34.101.31 (пункт 6.8.3). Алгоритм берет на вход транспортируемый ключ $X \in \{0,1\}^{8*}$, заголовок $I \in \{0,1\}^{128}$, ключ защиты $\theta \in \{0,1\}^{256}$ и возвращает защищенный ключ $Y \in \{0,1\}^{|X|+128}$.

Алгоритм `belt-keyunwrap`. Используется алгоритм `belt-keyunwrap`, определенный в СТБ 34.101.31 (пункт 6.8.4). Алгоритм берет на вход защищенный ключ $Y \in \{0,1\}^*$, заголовок $I \in \{0,1\}^{128}$, ключ защиты $\theta \in \{0,1\}^{256}$ и возвращает либо признак `ОШИБКА`, либо транспортируемый ключ $X \in \{0,1\}^{|Y|-128}$. Возврат признака `ОШИБКА` означает нарушение целостности транспортируемого ключа.

Одноразовый личный ключ k. При создании токена используется одноразовый личный ключ $k \in \{1,2,\ldots,q-1\}$. Требования по управлению k определены в 5.4.

Переменная θ. Используется переменная $\theta \in \{0,1\}^{256}$. Значение θ должно быть уничтожено после использования.

Переменная R. Используется переменная $R = (x_R, y_R) \in E^*_{a,b}(\mathbb{F}_p)$.

7.2.3 Алгоритм создания токена ключа

Алгоритм создания токена ключа состоит в выполнении следующих шагов:

1 Выработать $k \xleftarrow{R} \{1,2,\ldots,q-1\}$ (в соответствии с требованиями пункта 5.4).

2 Установить $R \leftarrow kG$.

3 Установить $\theta \leftarrow \langle kQ \rangle_{256}$.

4 Установить $Y \leftarrow \langle R \rangle_{2l} \parallel \texttt{belt-keywrap}(X, I, \theta)$.

5 Возвратить Y.

7.2.4 Алгоритм разбора токена ключа

Алгоритм разбора токена ключа состоит в выполнении следующих шагов:

1 Если длина Y не кратна 8 или $|Y| < 2l + 256$, то возвратить ОШИБКА.

2 Представить Y в виде $Y_0 \parallel Y_1$, где $Y_0 \in \{0,1\}^{2l}$, $Y_1 \in \{0,1\}^{|Y|-2l}$.

3 Установить $x_R \leftarrow \overline{Y}_0$.

4 Если $x_R \geqslant p$, то возвратить ОШИБКА.

5 Установить $y_R \leftarrow (x_R^3 + ax_R + b)^{(p+1)/4} \bmod p$.

6 Если $y_R^2 \not\equiv x_R^3 + ax_R + b \pmod p$, то возвратить ОШИБКА.

7 Построить $R = (x_R, y_R)$.

8 Установить $\theta \leftarrow \langle dR \rangle_{256}$.

9 Если $\texttt{belt-keyunwrap}(Y_1, I, \theta) = $ ОШИБКА, то возвратить ОШИБКА.

10 Установить $X \leftarrow \texttt{belt-keyunwrap}(Y_1, I, \theta)$.

11 Возвратить X.

Приложение А

(справочное)

Кодирование идентификаторов объектов

Пусть D — некоторый объект, снабженный идентификатором $\{d_1\ d_2\ \ldots d_n\}$ в соответствии с ГОСТ 34.973. Допустимый идентификатор должен удовлетворять следующим ограничениям: $d_1, d_2, \ldots, d_n$ — неотрицательные целые числа; $n \geqslant 2$; $d_1 \in \{0, 1, 2\}$; если $d_1 \in \{0, 1\}$, то $d_2 < 40$.

Для определения $\mathrm{OID}(D)$ числа $40d_1 + d_2, d_3, \ldots, d_n$ кодируются двоичными словами, которые последовательно конкатенируются и образуют составное слово V. Каждое кодируемое число d записывается в виде

$$d = \sum_{j=0}^{r} a_j 128^j, \quad 0 \leqslant a_j < 128,$$

где $a_r \neq 0$, если $d \neq 0$, и $r = a_0 = 0$ при $d = 0$. Затем число d кодируется словом

$$\langle 128 + a_r \rangle_8 \,\|\, \langle 128 + a_{r-1} \rangle_8 \,\|\, \ldots \,\|\, \langle 128 + a_1 \rangle_8 \,\|\, \langle a_0 \rangle_8.$$

После определения V вычисляется его длина $l = |V|/8$ в октетах. Если $l < 128$, то длина кодируется словом $L = \langle l \rangle_8$. Если $l \geqslant 128$, то длина представляется в виде

$$l = \sum_{j=0}^{r} b_j 256^j, \quad 0 \leqslant b_j < 256, \quad b_r \neq 0,$$

и кодируется словом

$$L = \langle 128 + r \rangle_8 \,\|\, \langle b_r \rangle_8 \,\|\, \langle b_{r-1} \rangle_8 \,\|\, \ldots \,\|\, \langle b_0 \rangle_8.$$

Окончательно $\mathrm{OID}(D)$ определяется как

$$\mathrm{OID}(D) = \mathtt{06}_{16} \,\|\, L \,\|\, V.$$

Например, идентификатору {1 2 112 0 2 0 34 101 31 81} соответствуют числа 42, 112, 0, 2, 0, 34, 101, 31 и 81. Данные числа кодируются словами $\mathtt{2A}_{16}$, $\mathtt{70}_{16}$, $\mathtt{00}_{16}$, $\mathtt{02}_{16}$, $\mathtt{00}_{16}$, $\mathtt{22}_{16}$, $\mathtt{65}_{16}$, $\mathtt{1F}_{16}$ и $\mathtt{51}_{16}$, которые образуют слово $V = \mathtt{2A7000020022651F51}_{16}$. Длина V кодируется словом $L = \mathtt{09}_{16}$. Окончательно кодовое представление исходного идентификатора имеет следующий вид: $\mathtt{06092A7000020022651F51}_{16}$.

Приложение Б

(рекомендуемое)

Стандартные параметры эллиптической кривой

В таблицах Б.1 — Б.3 представлены стандартные параметры эллиптической кривой для различных уровней стойкости.

Таблица Б.1 — Стандартные параметры ($l = 128$)

p	$2^{256} - 189$
$\langle p \rangle_{256}$	43FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
a	$2^{256} - 192$
$\langle a \rangle_{256}$	40FFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle b \rangle_{256}$	F1039CD6 6B7D2EB2 53928B97 6950F54C BEFBD8E4 AB3AC1D2 EDA8F315 156CCE77$_{16}$
$seed$	5E380100 00000000$_{16}$
q	$2^{256} - 51\ 359303463\ 308904523\ 350978545\ 619999225$
$\langle q \rangle_{256}$	07663D26 99BF5A7E FC4DFB0D D68E5CD9 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle y_G \rangle_{256}$	936A5104 18CF291E 52F608C4 66399178 5D83D651 A3C9E45C 9FD616FB 3CFCF76B$_{16}$

Таблица Б.2 — Стандартные параметры ($l = 192$)

p	$2^{384} - 317$
$\langle p \rangle_{384}$	C3FEFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
a	$2^{384} - 320$
$\langle a \rangle_{384}$	C0FEFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle b \rangle_{384}$	64BF7368 23FCA7BC 7CBDCEF3 F0E2BD14 3A2E71E9 F96A21A6 96B1FB0F BB482771 D2345D65 AB5A0733 20EF9C95 E1DF753C$_{16}$
$seed$	23AF0000 00000000$_{16}$
q	$2^{384} - 9886\ 438520659\ 958522437\ 788006980\ 660965037\ 549058207\ 958390857$
$\langle q \rangle_{384}$	B7A70CF3 3FDCB73D 0AFFA4A6 E7DA4680 BB7BAF73 03C4CC6C FEFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle y_G \rangle_{384}$	51C433F7 31CB5EEA F9422A6B 273E4084 55D3B166 9EE74905 A0FF86DC 119A723A 89BF2D43 7E113063 9E9E2EA8 2482435D$_{16}$

Таблица Б.3 — Стандартные параметры ($l = 256$)

p	$2^{512} - 569$
$\langle p \rangle_{512}$	C7FDFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
a	$2^{512} - 572$
$\langle a \rangle_{512}$	C4FDFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle b \rangle_{512}$	909C13D6 98693409 7AA2493A 272286EA 43A2AC87 8C003329 955E24C4 B5DC1127 88B0ADDA E313CE17 51255DDD EEA9C65B 8958FD60 6A5D8CD8 438C3B93 4459B46C$_{16}$
$seed$	AE170200 00000000$_{16}$
$\langle q \rangle_{512}$	F18E060D 49ADFFDC 32DF5695 E5CA1B36 F413212E B0EB6BF2 4E009801 2C09C0B2 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF$_{16}$
$\langle y_G \rangle_{512}$	BDEDEFCE 6FAE92B7 040D4CC9 B983AA67 6122E8EE 957377FF D26FFA0E E2DD7369 DACACC00 1BF8EDD2 E2BC61B3 B341ABB0 AB8FD1A0 F7E682B1 817603E4 7AFF26A8$_{16}$

Приложение В

(рекомендуемое)

Идентификационная электронная цифровая подпись

В.1 Назначение

Концепция идентификационной ЭЦП (далее — ИЭЦП), предложенная в работе [5], может быть реализована на основе алгоритмов настоящего стандарта, как это описывается в настоящем приложении.

Проверка ИЭЦП объединяет контроль целостности и подлинности сообщения с контролем подлинности подписавшей его стороны. Для сравнения, в инфраструктурах открытых ключей на основе сертификатов СТБ 34.101.19 требуется проверять ЭЦП, по-крайней мере, дважды: сначала подпись сертификата, а уже затем непосредственно подпись сообщения.

Для организации ИЭЦП требуется наличие доверенной стороны. Доверенная сторона вырабатывает личный ключ d и соответствующий открытый ключ Q с помощью алгоритма 6.2.2. Остальные стороны получают свои ключи от доверенной стороны.

Стороны снабжаются уникальными идентификаторами. Сторона с идентификатором Id обращается к доверенной стороне с запросом на получение ключей. Доверенная сторона проверяет подлинность Id, подписывает Id на ключе d с помощью алгоритма 7.1.3 и возвращает подпись обратно.

Подпись позволяет определить личный ключ обратившейся стороны и поэтому должна возвращаться конфиденциально. Доверенная сторона должна контролировать подписываемые ею данные и не должна раскрывать подписи сообщений, которые могут быть идентификаторами.

Сторона с идентификатором Id проверяет полученную подпись на ключе Q и одновременно извлекает из нее свои личный ключ e и открытый ключ R, используя алгоритм В.2.3. Доверенная сторона также может определить эти ключи и, таким образом, ИЭЦП сохраняет свойства обычной ЭЦП только тогда, когда доверенная сторона не использует e для подписи сообщений от чужого имени.

После извлечения ключей можно подписывать сообщения. Подпись вырабатывается с помощью алгоритма В.2.4, а проверяется с помощью алгоритма В.2.5. При выработке подписи используются идентификатор Id и личный ключ e. При проверке подписи используются открытые ключи Q и R, а также идентификатор Id. Подпись передается и хранится вместе с открытым ключом R (см. В.3).

Личный ключ e разрешено использовать только в алгоритме выработки ИЭЦП (в том числе для генерации одноразового ключа, см. В.2.2).

В.2 Алгоритмы

В.2.1 Входные и выходные данные

Входными данными алгоритмов ИЭЦП являются параметры p, a, b, q, G, которые описывают группу точек эллиптической кривой. Параметры должны удовлетворять условиям алгоритма 6.1.4. По модулю p определяется уровень стойкости l как минимальное натуральное число, для которого $p < 2^{2l}$.

Кроме параметров эллиптической кривой, входными данными алгоритма извлечения пары ключей являются идентификатор $Id \in \{0,1\}^*$, подпись $S \in \{0,1\}^*$ идентификатора, выработанная доверенной стороной, и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$ доверенной стороны. Открытый ключ Q должен удовлетворять условиям алгоритма 6.2.3.

Выходными данными алгоритма извлечения пары ключей является либо признак `ОШИБКА`, либо пара ключей: личный ключ $e \in \{0,1,\dots,q-1\}$ и открытый ключ $R \in E^*_{a,b}(\mathbb{F}_p)$. Возврат признака ошибка означает, что S не является корректной подписью Id и ключи не могут быть из нее извлечены.

Кроме параметров эллиптической кривой, входными данными алгоритма выработки ИЭЦП являются идентификатор $Id \in \{0,1\}^*$, сообщение $X \in \{0,1\}^*$ и личный ключ $e \in \{0,1,\dots,q-1\}$.

Выходными данными алгоритма выработки ИЭЦП является слово $S \in \{0,1\}^{3l}$ — идентификационная подпись X.

Кроме параметров эллиптической кривой, входными данными алгоритма проверки ИЭЦП являются идентификатор $Id \in \{0,1\}^*$, сообщение $X \in \{0,1\}^*$, идентификационная подпись $S \in \{0,1\}^*$, открытый ключ $R = (x_R, y_R)$,где x_R, y_R — целые числа, и открытый ключ $Q \in E^*_{a,b}(\mathbb{F}_p)$ доверенной стороны. Открытый ключ Q должен удовлетворять условиям алгоритма 6.2.3.

Выходными данными алгоритма проверки ИЭЦП является ответ `ДА` или `НЕТ`. Ответ `ДА` означает, что S является корректной подписью X, выработанной стороной с идентификатором Id. Ответ `НЕТ` означает обратное.

В.2.2 Вспомогательные алгоритмы и преобразования, переменные

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, описанный в 6.1.2.

Проверка точки эллиптической кривой. На шаге 4 алгоритма проверки ИЭЦП контролируется, что открытый ключ R является аффинной точкой эллиптической кривой. Контроль может быть выполнен с помощью алгоритма проверки открытого ключа, определенного в 6.2.3.

Функция h. Используется функция хэширования h, которая действует из $\{0,1\}^*$ в $\{0,1\}^{2l}$. Требования к h определены в 5.5.

Одноразовый личный ключ k. При выработке ИЭЦП используется одноразовый личный ключ $k \in \{1,2,\dots,q-1\}$. Требования по управлению k определены в 5.4. Разрешается вырабатывать k с помощью алгоритма 6.3.3 при замене d на e и с синхропосылкой H, полученной на шаге 2 алгоритма В.2.4.

Переменные H_0, H. Используются переменные $H_0, H \in \{0,1\}^{2l}$.

Переменная V. Используется переменная $V \in E_{a,b}(\mathbb{F}_p)$.

Переменная t. При проверке ИЭЦП используется переменная $t \in \{0,1\}^l$.

В.2.3 Алгоритм извлечения пары ключей

Извлечение пары ключей состоит в выполнении следующих шагов:

1 Если $|S| \neq 3l$, то возвратить `ОШИБКА`.

2 Представить S в виде $S = S_0 \parallel S_1$, где $S_0 \in \{0,1\}^l$, $S_1 \in \{0,1\}^{2l}$.

3 Если $\overline{S}_1 \geqslant q$, то возвратить ОШИБКА.
4 Установить $H_0 \leftarrow h(Id)$.
5 Установить $e \leftarrow (\overline{S}_1 + \overline{H}_0) \bmod q$.
6 Установить $V \leftarrow eG + (\overline{S}_0 + 2^l)Q$.
7 Если $V = O$, то возвратить ОШИБКА.
8 Установить $t \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle V \rangle_{2l} \parallel H_0) \big\rangle_l$.
9 Если $S_0 \neq t$, то возвратить ОШИБКА.
10 Установить $R \leftarrow V$.
11 Возвратить (e, R).

В.2.4 Алгоритм выработки идентификационной электронной цифровой подписи

ИЭЦП составляется из частей $S_0 \in \{0,1\}^l$ и $S_1 \in \{0,1\}^{2l}$. Выработка ИЭЦП состоит в выполнении следующих шагов:

1 Установить $H_0 \leftarrow h(Id)$.
2 Установить $H \leftarrow h(X)$.
3 Выработать $k \xleftarrow{R} \{1, 2, \ldots, q-1\}$ (в соответствии с требованиями пункта 5.4).
4 Установить $V \leftarrow kG$.
5 Установить $S_0 \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle V \rangle_{2l} \parallel H_0 \parallel H) \big\rangle_l$.
6 Установить $S_1 \leftarrow \big\langle (k - \overline{H} - (\overline{S}_0 + 2^l)e) \bmod q \big\rangle_{2l}$.
7 Установить $S \leftarrow S_0 \parallel S_1$.
8 Возвратить S.

В.2.5 Алгоритм проверки идентификационной электронной цифровой подписи

Проверка ИЭЦП состоит в выполнении следующих шагов:

1 Если $|S| \neq 3l$, то возвратить НЕТ.
2 Представить S в виде $S = S_0 \parallel S_1$, где $S_0 \in \{0,1\}^l$, $S_1 \in \{0,1\}^{2l}$.
3 Если $\overline{S}_1 \geqslant q$, то возвратить НЕТ.
4 Проверить, что $R \in E^*_{a,b}(\mathbb{F}_p)$. Если условие не выполняется, то возвратить НЕТ.
5 Установить $H_0 \leftarrow h(Id)$.
6 Установить $H \leftarrow h(X)$.
7 Установить $t \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle R \rangle_{2l} \parallel H_0) \big\rangle_l$.
8 Установить $V \leftarrow \big((\overline{S}_1 + \overline{H}) \bmod q\big) G + (\overline{S}_0 + 2^l)(R - (\overline{t} + 2^l)Q)$.
9 Если $V = O$, то возвратить НЕТ.
10 Установить $t \leftarrow \big\langle \texttt{belt-hash}(\mathrm{OID}(h) \parallel \langle V \rangle_{2l} \parallel H_0 \parallel H) \big\rangle_l$.
11 Если $S_0 \neq t$, то возвратить НЕТ.
12 Возвратить ДА.

В.3 Объединение идентификационной электронной цифровой подписи с открытым ключом

При передаче и хранении ИЭЦП $S \in \{0,1\}^{3l}$ должна объединяться с открытым ключом $R \in E^*_{a,b}(\mathbb{F}_p)$ в виде слова $S \parallel \langle R \rangle_{4l}$.

Приложение Г
(справочное)
Проверочные примеры

Г.1 Генерация пары ключей

В таблице Г.1 представлен пример генерации личного ключа. Используются параметры эллиптической кривой, заданные в таблице Б.1.

Таблица Г.1 — Генерация пары ключей

$\langle d \rangle_{256}$	1F66B5B8 4B733967 4533F032 9C74F218 34281FED 0732429E 0C79235F C273E269$_{16}$
$\langle x_Q \rangle_{256}$	BD1A5650 179D79E0 3FCEE49D 4C2BD5DD F54CE46D 0CF11E4F F87BF7A8 90857FD0$_{16}$
$\langle y_Q \rangle_{256}$	7AC6A603 61E8C817 3491686D 461B2826 190C2EDA 5909054A 9AB84D2A B9D99A90$_{16}$

Г.2 Выработка электронной цифровой подписи

В таблице Г.2 представлен пример выработки ЭЦП. Используются параметры эллиптической кривой, заданные в таблице Б.1, и личный ключ, заданный в таблице Г.1. В качестве h используется функция, заданная алгоритмом `belt-hash`.

Таблица Г.2 — Выработка электронной цифровой подписи

X	B194BAC8 0A08F53B 366D008E 58$_{16}$
H	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$
$\langle k \rangle_{256}$	4C0E74B2 CD5811AD 21F23DE7 E0FA742C 3ED6EC48 3C461CE1 5C33A77A A308B7D2$_{16}$
$\langle x_R \rangle_{256}$	CCEEF1A3 13A40664 9D15DA0A 851D486A 695B641B 20611776 252FFDCE 39C71060$_{16}$
$\langle y_R \rangle_{256}$	7C9EA1F3 3C23D20D FCB8485A 88BE6523 A28ECC32 15B47FA2 89D6C9BE 1CE837C0$_{16}$
S_0	E36B7F03 77AE4C52 4027C387 FADF1B20$_{16}$
S_1	CE72F153 0B71F2B5 FD3A8C58 4FE2E1AE D20082E3 0C8AF650 11F4FB54 649DFD3D$_{16}$
S	E36B7F03 77AE4C52 4027C387 FADF1B20 CE72F153 0B71F2B5 FD3A8C58 4FE2E1AE D20082E3 0C8AF650 11F4FB54 649DFD3D$_{16}$

Г.3 Проверка электронной цифровой подписи

В таблице Г.3 представлен пример проверки ЭЦП. Используются параметры эллиптической кривой, заданные в таблице Б.1, и открытый ключ, заданный в таблице Г.1. В качестве h используется функция, заданная алгоритмом `belt-hash`.

Таблица Г.3 — Проверка ЭЦП

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
S	47A63C8B 9C936E94 B5FAB3D9 CBD78366 290F3210 E163EEC8 DB4E921E 8479D413 8F112CC2 3E6DCE65 EC5FF21D F4231C28$_{16}$
H	9D02EE44 6FB6A29F E5C982D4 B13AF9D3 E90861BC 4CEF27CF 306BFB0B 174A154A$_{16}$
S_0	47A63C8B 9C936E94 B5FAB3D9 CBD78366$_{16}$
S_1	290F3210 E163EEC8 DB4E921E 8479D413 8F112CC2 3E6DCE65 EC5FF21D F4231C28$_{16}$
$\langle x_R \rangle_{256}$	1D5A382B 962D4ED0 6193258C A6DE535D 8FD7FACB 853171E9 32EF93B5 EE800120$_{16}$
$\langle y_R \rangle_{256}$	03DBB7B5 BD070363 80BAFA47 FCA7E6CA 3F179EDD D1AE5086 64790918 3628EDDC$_{16}$
t	47A63C8B 9C936E94 B5FAB3D9 CBD78366$_{16}$

Условие $S_0 = t$ выполняется, и алгоритм возвращает ДА.

Г.4 Создание токена ключа

В таблице Г.4 представлен пример создания токена ключа. Используются параметры эллиптической кривой, заданные в таблице Б.1, и открытый ключ, заданный в таблице Г.1.

Таблица Г.4 — Создание токена ключа

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504$_{16}$
I	5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
$\langle k \rangle_{256}$	0F51D913 47617C20 BD4AB07A EF4F26A1 AD1362A8 F9A3D42F BE1B8E6F 1C88AAD5$_{16}$
$\langle x_R \rangle_{256}$	9B4EA669 DABDF100 A7D4B6E6 EB76EE52 51912531 F426750A AC8A9DBB 51C54D8D$_{16}$
$\langle y_R \rangle_{256}$	6AB7DBF1 5FCBD768 EE68A173 F7B236EF C15A01E2 AA6CD1FE 98B947DA 7B38A2A0$_{16}$
θ	11B3A639 83BCCB6D 32C5943F 66F01D4C EA8CEE35 E4A6AE98 B1407C53 674317AC$_{16}$
Y	9B4EA669 DABDF100 A7D4B6E6 EB76EE52 51912531 F426750A AC8A9DBB 51C54D8D EB9289B5 0A46952D 0531861E 45A8814B 008FDC65 DE9FF1FA 2A1F16B6 A280E957 A814$_{16}$

Г.5 Разбор токена ключа

В таблице Г.5 представлен пример разбора токена ключа. Используются параметры эллиптической кривой, заданные в таблице Б.1, и личный ключ, заданный в таблице Г.1.

Таблица Г.5 — Разбор токена ключа

Y	4856093A 0F6C1301 5FC8E15F 1B23A762 02D2F4BA 6E5EC52B 78658477 F6486DE6 87AFAEEA 0EF7BC13 26A7DCE7 A10BA10E 3F91C012 6044B222 67BF30BD 6F1DA29E 0647CF39 C1D59A56 BB0194E0 F4F8A2BB$_{16}$
I	E12BDC1A E28257EC 703FCCF0 95EE8DF1$_{16}$
$\langle x_R \rangle_{256}$	4856093A 0F6C1301 5FC8E15F 1B23A762 02D2F4BA 6E5EC52B 78658477 F6486DE6$_{16}$
Y_1	87AFAEEA 0EF7BC13 26A7DCE7 A10BA10E 3F91C012 6044B222 67BF30BD 6F1DA29E 0647CF39 C1D59A56 BB0194E0 F4F8A2BB$_{16}$
$\langle y_R \rangle_{256}$	DA4FE935 574DA2F0 117AFE25 971DFD62 9D985CE9 E4F1052C 66445686 2C83CD37$_{16}$
θ	3E2D4915 38A58FA5 108CF809 85222670 661794AB 2423E410 9E785A22 D1529BC6$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D$_{16}$

Г.6 Генерация одноразового личного ключа

В таблицах Г.6, Г.7 представлены примеры генерации одноразового личного ключа. Используется число q, заданное в таблице Б.1, и личный ключ d, заданный в таблице Г.1. В качестве h используется функция, заданная алгоритмом `belt-hash`.

Таблица Г.6 — Генерация одноразового ключа (t — пустое слово)

H	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$
θ	D61E3A91 0550E3BC AD5BF4F5 26FB8DAA DEA9C132 E0BAEE03 169DF4DF 9BD6C20C$_{16}$
$\langle k \rangle_{256}$	829614D8 411DBBC4 E1F2471A 40045864 40FD8C95 53FAB6A1 A45CE417 AE97111E$_{16}$

Таблица Г.7 — Генерация одноразового ключа

H	9D02EE44 6FB6A29F E5C982D4 B13AF9D3 E90861BC 4CEF27CF 306BFB0B 174A154A$_{16}$
t	BE329713 43FC9A48 A02A885F 194B09A1 7ECDA4D0 1544AF$_{16}$
θ	AE443163 32A85C3B 9F6B31EE EADFF088 D30FE507 021AC86A 3EC8E087 4ED33648$_{16}$
$\langle k \rangle_{256}$	7ADC8713 283EBFA5 47A2AD9C DFB245AE 0F7B968D F0F91CB7 85D1F932 A3583107$_{16}$

Г.7 Идентификационная электронная цифровая подпись

В таблицах Г.8 — Г.10 представлены примеры извлечения пары ключей, выработки и проверки ИЭЦП. Используются параметры эллиптической кривой, заданные в таблице Б.1, и открытый ключ Q, заданный в таблице Г.1. В качестве h используется функция, заданная алгоритмом `belt-hash`. Ключи e, R из таблицы Г.8 используются в примерах из таблиц Г.9, Г.10.

Таблица Г.8 — Извлечение пары ключей

Id	B194BAC8 0A08F53B 366D008E 58_{16}
S	E36B7F03 77AE4C52 4027C387 FADF1B20 CE72F153 0B71F2B5 FD3A8C58 4FE2E1AE D20082E3 0C8AF650 11F4FB54 649DFD3D$_{16}$
$\langle e \rangle_{256}$	79628979 DF369BEB 94DEF329 9476AED4 14F39148 AA69E31A 7397E8AA 70578AB3$_{16}$
$\langle x_R \rangle_{256}$	CCEEF1A3 13A40664 9D15DA0A 851D486A 695B641B 20611776 252FFDCE 39C71060$_{16}$
$\langle y_R \rangle_{256}$	7C9EA1F3 3C23D20D FCB8485A 88BE6523 A28ECC32 15B47FA2 89D6C9BE 1CE837C0$_{16}$

Таблица Г.9 — Выработка ИЭЦП

Id	B194BAC8 0A08F53B 366D008E 58_{16}
X	5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
H_0	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$
H	8CAC3CF6 7106FFEF 54C55DA7 65D207A5 D6B47FD6 9B894DF1 A17DC067 608B9362$_{16}$
$\langle k \rangle_{256}$	C431B41B BE8E8022 88737ACF 45A29251 FC736A3C 6F478F77 A7ED271D 5EEDAA58$_{16}$
$\langle x_V \rangle_{256}$	88573C0E A51BC6EE 2EAC6EF1 BD74B862 FD73805C EAD484E3 C4FE02C5 BF2056EA$_{16}$
$\langle y_V \rangle_{256}$	FAFA4DF2 2B69D948 8B310A4B A3BEACE9 CA5D8E5C D187790F D9357B34 7D4B17E5$_{16}$
S_0	1697FE6A 073D3B28 C9D0DD83 2A169D7B$_{16}$
S_1	8D342FDC 47BC8AAE B6226448 956E22D6 CC73B62C B21B66E5 C8DE0A3E 234FB0C6$_{16}$
S	1697FE6A 073D3B28 C9D0DD83 2A169D7B 8D342FDC 47BC8AAE B6226448 956E22D6 CC73B62C B21B66E5 C8DE0A3E 234FB0C6$_{16}$

Таблица Г.10 — Проверка ИЭЦП

Id	B194BAC8 0A08F53B 366D008E 58_{16}
X	5BE3D612 17B96181 FE6786AD 716B890B 5CB0C0FF 33C356$_{16}$
S	31CBA14F C2D79AFC D8F50E29 F993FC2C B270BD0A 79D534B3 B1207914 00C8BB18 50AD6D3C 78047FCB 46F18608 AC7006AA$_{16}$
H_0	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$
H	3A443648 6039352C BE3370D7 A899F5FA DB5E583D B8A49FB2 7DCC588F 32D6F344$_{16}$
$\langle k \rangle_{256}$	C431B41B BE8E8022 88737ACF 45A29251 FC736A3C 6F478F77 A7ED271D 5EEDAA58$_{16}$
$\langle x_V \rangle_{256}$	8B3A7766 33C119A0 C6EE9828 BCE4FC56 A806A547 18FAB30B 3DB3B564 39FACAF4$_{16}$
$\langle y_V \rangle_{256}$	26EAF9E5 6A8C0E1F ABDB7114 E10594E4 7DE490C4 45998002 B8E26D2C AABF91AC$_{16}$
t (шаг 10)	31CBA14F C2D79AFC D8F50E29 F993FC2C$_{16}$

Приложение Д
(рекомендуемое)
Модуль ACH.1

Д.1 Идентификаторы

Алгоритмам стандарта присваиваются следующие идентификаторы:

`bign-with-hspec`	алгоритмы ЭЦП (7.1) с функцией хэширования, определяемой долговременными параметрами;
`bign-with-hbelt`	алгоритмы ЭЦП (7.1) с функцией хэширования, заданной алгоритмом `belt-hash`;
`bign-genec`	алгоритм генерации параметров эллиптической кривой (6.1.3);
`bign-valec`	алгоритм проверки параметров эллиптической кривой (6.1.4);
`bign-genkeypair`	алгоритм генерации пары ключей (6.2.2);
`bign-valpubkey`	алгоритм проверки открытого ключа (6.2.3);
`bign-keytransport`	алгоритмы транспорта ключа (7.2);
`bign-genk`	алгоритм генерации одноразового личного ключа (6.3.3);
`bign-ibs-with-hspec`	алгоритмы ИЭЦП (В.2) с функцией хэширования, определяемой долговременными параметрами;
`bign-ibs-with-hbelt`	алгоритмы ИЭЦП (В.2) с функцией хэширования, заданной алгоритмом `belt-hash`.

Уровень стойкости алгоритмов ЭЦП, транспорта ключа и ИЭЦП не указывается в их идентификаторах, а определяется по размерностям параметров используемой эллиптической кривой. Размерности параметров и длина значений используемой функции хэширования должны соответствовать друг другу.

Открытому ключу, который вырабатывается по алгоритму 6.2.2, присваивается идентификатор `bign-pubkey`. Открытый ключ может использоваться в алгоритмах ЭЦП и (или) транспорта ключа.

Стандартным параметрам эллиптической кривой, заданным в приложении Б, присваиваются следующие идентификаторы:

`bign-curve256v1`	параметры, определенные в таблице Б.1;
`bign-curve384v1`	параметры, определенные в таблице Б.2;
`bign-curve512v1`	параметры, определенные в таблице Б.3.

Д.2 Параметры алгоритмов

Идентификаторы алгоритмов ЭЦП, транспорта ключа и ИЭЦП могут указываться в компоненте `algorithm` следующего типа ACH.1:

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

Этот тип описывает, например, компоненты с именами `signatureAlgorithm` и `signature` в определениях форматов запроса на получение сертификата (СТБ 34.101.17), сертификатов и списка отозванных сертификатов (СТБ 34.101.19), подписанных данных (СТБ 34.101.23), запроса и ответа о статусе сертификата (СТБ 34.101.26). Во всех перечисленных случаях если в `algorithm` установлен идентификатор `bign-with-hspec`, то в `parameters` должен быть задан идентификатор используемой функции хэширования h. Если же в `algorithm` установлен идентификатор `bign-with-hbelt`, то `parameters` должен равняться `NULL`, а h определяться алгоритмом `belt-hash`.

Тип `AlgorithmIdentifier` описывает также компоненты с именами `keyEncryptionAlgorithm` в определениях форматов конвертованных, аутентифицируемых и аутентифицируемых конвертованных данных (СТБ 34.101.23). Эти компоненты определяют способ защиты ключей шифрования и имитозащиты данных. Если для защиты используются алгоритмы транспорта ключа (в `RecipientInfo` выбран компонент `ktri`, см. пункт 9.3 СТБ 34.101.23), а в `algorithm` установлен идентификатор `bign-keytransport`, то `parameters` должен равняться `NULL`, а заголовок I транспортируемого ключа полагаться равным 0^{128}.

Д.3 Описание двоичных слов

Слово $u \in \{0,1\}^{8*}$ может описываться строкой битов (`BIT STRING`), последовательными (от первого к замыкающему, см. пункт 15 ГОСТ 34.973) элементами которой являются последовательные символы u. Слово u может также описываться строкой октетов (`OCTET STRING`), составленной из последовательных октетов u.

Указанные описания u согласованы между собой. Например, при кодировании u по правилам ГОСТ 34.974 содержимое u как `BIT STRING` отличается от содержимого u как `OCTET STRING` только нулевым октетом-префиксом.

Д.4 Описание конечного поля и его элементов

Для описания конечного поля, над которым строится эллиптическая кривая, используется тип

```
FieldID ::= SEQUENCE {
  fieldType   OBJECT IDENTIFIER (bign-primefield),
  parameters  INTEGER
}
```

Компонент `fieldType` этого типа определяет вид поля. Примененный синтаксис обязывает использовать только простые конечные поля, которым назначен идентификатор `bign-primefield`. Компонент `parameters` описывает модуль p поля $\mathbb{F}_p$.

На уровне стойкости l элемент u поля $\mathbb{F}_p$ должен представляться строкой октетов как двоичное слово $\langle u \rangle_{2l}$.

Д.5 Описание параметров эллиптической кривой

Параметры эллиптической кривой могут представляться значениями типа

```
DomainParameters ::= CHOICE {
```

```
  specified  ECParameters,
  named      OBJECT IDENTIFIER,
  implicit   NULL
}
```

Выбор компонента `specified` означает явное задание параметров. Компонент `named` используется для ссылки на именованные параметры, заданные в приложении А или в другом документе. Компонент `implicit` используется для указания на то, что наследуются параметры внешнего источника, например удостоверяющего центра.

Явно задаваемые параметры представляются значениями типа

```
ECParameters ::= SEQUENCE {
  version  INTEGER {ecpVer1(1)} (ecpVer1),
  fieldID  FieldID,
  curve    Curve,
  base     OCTET STRING (SIZE(32|48|64)),
  order    INTEGER,
  cofactor INTEGER (1) OPTIONAL
}
```

Компонент `version` указывает на версию данного типа АСН.1. Примененный синтаксис обязывает использовать версию 1, которая обозначена через `ecpVer1`. Компонент `fieldID` описывает поле $\mathbb{F}_p$, над которым строится эллиптическая кривая. Компонент `curve` описывает уравнение эллиптической кривой. Компонент `base` описывает базовую точку эллиптической кривой. Компонент `order` описывает порядок q группы точек эллиптической кривой. Необязательный компонент `cofactor` описывает отношение порядка всей группы точек эллиптической кривой к порядку группы, порожденной базовой точкой. Это отношение (кофактор) для эллиптических кривых настоящего стандарта всегда равняется 1.

Для описания уравнения эллиптической кривой используется тип

```
Curve ::= SEQUENCE {
  a     OCTET STRING (SIZE(32|48|64)),
  b     OCTET STRING (SIZE(32|48|64)),
  seed  BIT STRING (SIZE(64))
}
```

Последовательные компоненты этого типа определяют коэффициенты a, b эллиптической кривой и параметр *seed*, использованный для построения b при заданных p и a.

Коэффициенты a, b и базовая точка $G = (0, y_G)$ задаются строками октетов. Эти строки строятся по a, b и y_G как элементам поля $\mathbb{F}_p$ по правилам, определенным в Д.4. Нулевая x-координата точки G опускается.

Д.6 Описание открытого ключа

На уровне стойкости l открытому ключу Q ставится в соответствие двоичное слово $\langle Q \rangle_{4l}$, для описания которого может использоваться тип

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

В запросе на получение сертификата (СТБ 34.101.17) и в сертификатах (СТБ 34.101.19) открытый ключ должен представляться значениями типа

```
SubjectPublicKeyInfo ::= SEQUENCE {
  algorithm         AlgorithmIdentifier,
  subjectPublicKey  PublicKey
}
```

Компонент `algorithm` этого типа описывает свойства открытого ключа. Компонент `subjectPublicKey` описывает значение открытого ключа.

Для описания свойств открытого ключа в компоненте `algorithm`, вложенном в `AlgorithmIdentifier`, должен быть установлен идентификатор `bign-pubkey`, а в компоненте `parameters` — значение типа `DomainParameters`.

Д.7 Описание электронной цифровой подписи

На уровне стойкости l ЭЦП является двоичным словом длины $3l$, для описания которого могут использоваться типы

```
Signature ::= BIT STRING (SIZE(384|576|768))
SignatureValue ::= OCTET STRING (SIZE(48|72|96))
```

В запросе на получение сертификата (СТБ 34.101.17), в сертификатах и списках отозванных сертификатов (СТБ 34.101.19), в запросе и ответе о статусе сертификата (СТБ 34.101.26) подпись должна представляться значением типа `Signature`, а в подписанных данных (СТБ 34.101.23) — значением типа `SignatureValue`.

Д.8 Описание токена ключа

В алгоритмах транспорта `bign-keytransport` токен ключа представляет собой двоичное слово, длина которого кратна 8.

В определениях форматов конвертованных, аутентифицируемых и аутентифицируемых конвертованных данных (СТБ 34.101.23) токен ключа должен устанавливаться в компоненте `encryptedKey` типа `KeyTransRecipientInfo` и описываться при этом строкой октетов.

Д.9 Описание идентификационной электронной цифровой подписи

На уровне стойкости l объединение ИЭЦП с открытым ключом, описанное в В.3, является двоичным словом длины $7l$. Для описания этого слова может использоваться тип

```
IdSignatureValue ::= OCTET STRING (SIZE(112|168|224))
```

Д.10 Совместимость

Введенные определения поддерживают использование алгоритмов настоящего стандарта при создании и обработке данных, форматы которых определены в СТБ 34.101.17, СТБ 34.101.19, СТБ 34.101.23, СТБ 34.101.26.

Тип `ECParameters` соответствует одноименному типу стандарта [2] со следующими уточнениями:

1 Простое конечное поле описывается идентификатором `bign-primefield` вместо идентификатора `{iso(1) member-body(2) us(840) 10045 4 1 field-type(1) 1}` в стандарте [2].

2 Элементы поля описываются строками октетов по правилам «от младших к старшим» (little-endian), принятым в настоящем стандарте, вместо правил «от старших к младшим» (big-endian), принятым в [2].

3 В компоненте `base` задается только y-координата базовой точки эллиптической кривой. Нулевая x-координата опускается.

Тип `DomainParameters` соответствует типу `Parameters` стандарта [2].

Д.11 Модуль ACH.1

```
Bign-module-v2 {iso(1) member-body(2) by(112) 0 2 0 34 101 45 module(1) ver2(2)}
DEFINITIONS ::=
BEGIN
  bign OBJECT IDENTIFIER ::= {iso(1) member-body(2) by(112) 0 2 0 34 101 45}

  bign-with-hspec OBJECT IDENTIFIER ::= {bign 11}
  bign-with-hbelt OBJECT IDENTIFIER ::= {bign 12}
  bign-genec OBJECT IDENTIFIER ::= {bign 21}
  bign-valec OBJECT IDENTIFIER ::= {bign 22}
  bign-genkeypair OBJECT IDENTIFIER ::= {bign 31}
  bign-valpubkey OBJECT IDENTIFIER ::= {bign 32}
  bign-keytransport OBJECT IDENTIFIER ::= {bign 41}
  bign-genk OBJECT IDENTIFIER ::= {bign 61}
  bign-ibs-with-hspec OBJECT IDENTIFIER ::= {bign 71}
  bign-ibs-with-hbelt OBJECT IDENTIFIER ::= {bign 72}

  bign-keys OBJECT IDENTIFIER ::= {bign keys(2)}
  bign-pubkey OBJECT IDENTIFIER ::= {bign-keys 1}

  bign-curves OBJECT IDENTIFIER ::= {bign curves(3)}
  bign-curve256v1 OBJECT IDENTIFIER ::= {bign-curves 1}
  bign-curve384v1 OBJECT IDENTIFIER ::= {bign-curves 2}
  bign-curve512v1 OBJECT IDENTIFIER ::= {bign-curves 3}

  bign-fields OBJECT IDENTIFIER ::= {bign fields(4)}
  bign-primefield OBJECT IDENTIFIER ::= {bign-fields prime(1)}
```

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

  DomainParameters ::= CHOICE {
    specified  ECParameters,
    named      OBJECT IDENTIFIER,
    implicit   NULL
  }

  ECParameters ::= SEQUENCE {
    version  INTEGER {ecpVer1(1)} (ecpVer1),
    fieldID  FieldID,
    curve    Curve,
    base     OCTET STRING (SIZE(32|48|64)),
    order    INTEGER,
    cofactor INTEGER (1) OPTIONAL
  }

  FieldID ::= SEQUENCE {
    fieldType   OBJECT IDENTIFIER (bign-primefield),
    parameters  INTEGER
  }

  Curve ::= SEQUENCE {
    a     OCTET STRING (SIZE(32|48|64)),
    b     OCTET STRING (SIZE(32|48|64)),
    seed  BIT STRING (SIZE(64))
  }

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

  SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm         AlgorithmIdentifier,
    subjectPublicKey  PublicKey
  }

  Signature ::= BIT STRING (SIZE(384|576|768))
  SignatureValue ::= OCTET STRING (SIZE(48|72|96))
  IdSignatureValue ::= OCTET STRING (SIZE(112|168|224))
END
```

Приложение Е

(справочное)

Парольная защита личного ключа

Е.1 Назначение

Конфиденциальность и контроль целостности личного ключа при его хранении могут обеспечиваться разными способами. Одним из таких способов является защита ключа на пароле владельца. В настоящем приложении определяются алгоритмы парольной защиты, идентификаторы алгоритмов, структуры данных для хранения параметров алгоритмов. Определения соответствуют стандарту [4].

Паролем является двоичное слово, длина которого кратна 8. Это слово может являться кодированным представлением обычной текстовой строки. Кодировать рекомендуется по правилам UTF-8, заданным в [3].

Парольная защита выполняется в два этапа. Сначала с помощью алгоритма Е.2.3 по паролю и дополнительным служебным данным строится ключ защиты θ. Затем на этом ключе с помощью алгоритмов, описанных в Е.3, устанавливается или снимается защита личного ключа.

Владелец личного ключа должен оградить доступ других лиц даже к защищенному ключу. Тем не менее, угроза такого доступа существует, и парольная защита организуется так, чтобы максимально затруднить определение ключа без знания пароля.

Для этого, во-первых, владелец должен использовать высокоэнтропийные пароли (большой длины, с цифрами и буквами, без многократных повторов символов и т.д.). Во-вторых, при построении θ выполняется несколько итераций, на которых последовательно вычисляются значения сложной необратимой функции. Регулируя число итераций c, можно сделать неприемлемо большим время, которое требуется злоумышленнику для перебора паролей, оставляя допустимым время, затрачиваемое владельцем на выработку ключа защиты. В-третьих, при построении θ кроме пароля используется синхропосылка S (salt, «соль» в [4]). Вырабатываемый ключ защиты зависит от выбранной синхропосылки и злоумышленник лишается возможности предварительно рассчитывать ключи для определенных классов паролей, т. е. проводить, так называемые, словарные атаки.

Число итераций c и синхропосылка S являются несекретными элементами и могут сохраняться вместе с защищенным личным ключом. Рекомендуется выбирать $c \geqslant 10000$ и использовать синхропосылки (двоичные слова), длина которых не меньше 64. Рекомендуется вырабатывать синхропосылки случайным или псевдослучайным методом.

Е.2 Построение ключа защиты по паролю

Е.2.1 Входные и выходные данные

Входными данными алгоритма построения ключа защиты являются пароль $P \in \{0,1\}^{8*}$, число итераций $c \in \{1,2,\ldots\}$ и синхропосылка $S \in \{0,1\}^{8*}$.

Выходными данными алгоритма является ключ защиты $\theta \in \{0,1\}^{256}$.

E.2.2 Вспомогательные алгоритмы

Используется алгоритм HMAC, определенный в СТБ 34.101.47 (пункт 6.1) с базовым алгоритмом хэширования `belt-hash`. Алгоритм $\text{HMAC}_{\texttt{belt-hash}}$ берет на вход ключ $K \in \{0,1\}^*$, сообщение $X \in \{0,1\}^*$ и возвращает слово $Y \in \{0,1\}^{256}$.

E.2.3 Алгоритм построения ключа защиты по паролю

Построение ключа защиты состоит в выполнении следующих шагов:

1 $\theta \leftarrow \text{HMAC}_{\texttt{belt-hash}}(P, S \parallel \texttt{00000001}_{16})$.

2 Для $i = 1, 2, \ldots, c$ выполнить:

  1) $\theta \leftarrow \text{HMAC}_{\texttt{belt-hash}}(P, \theta)$.

3 Возвратить θ.

E.3 Алгоритмы защиты личного ключа

Защита личного ключа состоит в применении алгоритмов `belt-keywrap` и `belt-keyunwrap`, описанных в 7.2.2.

В этих алгоритмах должен использоваться заголовок $I = 0^{128}$ и ключ $\theta \in \{0,1\}^{256}$, построенный по паролю. На уровне стойкости l личный ключ d должен представляться двоичным словом $\langle d \rangle_{2l}$.

Возврат алгоритмом `belt-keyunwrap` признака `ОШИБКА` означает, что ключ θ некорректен или что нарушена целостность личного ключа.

E.4 Идентификаторы и параметры алгоритмов

Алгоритмам парольной защиты присваиваются следующие идентификаторы:

`id-PBKDF2`	алгоритм построения ключа защиты по паролю (E.2.3);
`id-PBES2`	алгоритмы защиты личного ключа (E.3).

Данные идентификаторы определены в стандарте [4] следующим образом:

```
pkcs OBJECT IDENTIFIER ::= {iso(1) member-body(2) us(840) rsadsi(113549) 1}
pkcs-5 OBJECT IDENTIFIER ::= {pkcs 5}
id-PBKDF2 OBJECT IDENTIFIER ::= {pkcs-5 12}
id-PBES2 OBJECT IDENTIFIER ::= {pkcs-5 13}
```

Если идентификатор `id-PBKDF2` используется в компоненте `algorithm` типа `AlgorithmIdentifier` (см. Д.2), то соответствующий компонент `parameters` должен иметь тип

```
PBKDF2-params ::= SEQUENCE {
  salt CHOICE {
    specified OCTET STRING,
    otherSource AlgorithmIdentifier
  },
  iterationCount INTEGER (1..MAX),
  keyLength INTEGER (32) OPTIONAL,
  prf AlgorithmIdentifier
}
```

Компонент `salt` этого типа определяет синхропосылку S. Синхропосылка может быть задана явно строкой `specified`, либо алгоритмически через компонент `otherSource`. Компонент `iterationCount` описывает число итераций c. Необязательный компонент `keyLenght` описывает длину ключа защиты θ в октетах. Компонент `prf` описывает алгоритм $\text{HMAC}_{\texttt{belt-hash}}$.

Вложенный в `prf` компонент `algorithm` должен принимать значение `hmac-hbelt`, а компонент `parameters` — значение `NULL`. Идентификатор `hmac-hbelt` определен в СТБ 34.101.47 (приложение А).

Если идентификатор `id-PBES2` используется в компоненте `algorithm` типа `AlgorithmIdentifier`, то соответствующий компонент `parameters` должен иметь тип

```
PBES2-params ::= SEQUENCE {
  keyDerivationFunc AlgorithmIdentifier,
  encryptionScheme AlgorithmIdentifier
}
```

Вложенный в `keyDerivationFunc` компонент `algorithm` должен принимать значение `id-PBKDF2`, а компонент `parameters` — значение типа `PBKDF2-params`.

Вложенный в `encryptionScheme` компонент `algorithm` должен принимать значение `belt-keywrap256`, а компонент `parameters` — значение `NULL`.

Е.5 Проверочный пример

В таблице Е.1 представлен пример построения ключа защиты θ по паролю. В таблице дополнительно указывается результат Y защиты личного ключа d из таблицы Г.1 на ключе θ.

Таблица Е.1 — Построение ключа защиты по паролю

P	42313934 42414338 30413038 46353342_{16}
c	10000
S	BE329713 43FC9A48 A02A885F $194B09A1_{16}$
θ	D9024724 82130F3B 77D09303 03DD7E4E 68630CC0 2B56A8B2 AFA74F09 $6BCAC971_{16}$
Y	248E0CD7 639B1237 76F1CEC1 FCECE708 C2DFC53F 78ECEA6C 33B4C3C1 E6183AD6 D8A18CFA F540976E 1022B89D $BA32DA18_{16}$

Приложение Ж
(рекомендуемое)
Теоретико-числовые алгоритмы

Ж.1 Проверка простоты алгоритмом Рабина — Миллера

Ж.1.1 Входные и выходные данные

Входными данными алгоритма Рабина — Миллера являются натуральное нечетное число $n \geqslant 5$, простота которого проверяется, и число итераций T.

Выходными данными алгоритма является ответ ДА или НЕТ. Ответ ДА означает, что n вероятно простое. Ответ НЕТ означает, что n — составное.

Для простых n алгоритм всегда выдает верный ответ ДА. Для составных n может быть получен как верный ответ НЕТ, так и ошибочный ответ ДА. Вероятность ошибочного ответа уменьшается с ростом числа итераций и не превосходит 2^{-2T}. Если l — длина двоичного представления n, т. е. $2^{l-1} \leqslant n < 2^l$, то рекомендуется выбирать $T \geqslant l/4$.

Ж.1.2 Переменные

Используются переменные $u \in \{2, 3, \ldots, n-2\}$ и $v \in \{1, 2, \ldots, n-1\}$.

Ж.1.3 Алгоритм Рабина — Миллера

Проверка простоты n состоит в выполнении следующих шагов:

1 Представить n в виде $2^s r + 1$, где s — натуральное, r — натуральное нечетное.

2 Для $t = 1, 2, \ldots, T$ выполнить:
- 1) $u \xleftarrow{R} \{2, 3, \ldots, n-2\}$;
- 2) $v \leftarrow u^r \bmod n$;
- 3) если $v = 1$ или $v = n-1$, то перейти к шагу 2.6;
- 4) для $i = 1, 2, \ldots, s-1$ выполнить:
  - (a) $v \leftarrow v^2 \bmod n$;
  - (b) если $v = 1$, то возвратить НЕТ;
  - (c) если $v = n-1$, то перейти к шагу 2.6;
- 5) возвратить НЕТ;
- 6) продолжить.

3 Возвратить ДА.

Ж.2 Вычисление символа Лежандра

Ж.2.1 Входные и выходные данные

Входными данными алгоритма вычисления символа Лежандра являются нечетное простое число n и $u \in \{0, 1, \ldots, n-1\}$.

Выходными данными является символ Лежандра $\left(\frac{u}{n}\right)$.

Ж.2.2 Переменные

Входные данные интерпретируются как начальные значения переменных n и u. На шагах алгоритма эти значения изменяются, уменьшаясь.

Используется переменная $t \in \{-1, 1\}$.

Ж.2.3 Алгоритм вычисления символа Лежандра

Вычисление $\left(\frac{u}{n}\right)$ состоит в выполнении следующих шагов:

1 Установить $t \leftarrow 1$.

2 Если $u = 0$, то возвратить 0.

3 Если $u = 1$, то возвратить t.

4 Представить u в виде $2^s r$, где s — неотрицательное целое, r — натуральное нечетное.

5 Если s — нечетное, то

   1) если $n \equiv 3 \pmod 8$ или $n \equiv 5 \pmod 8$, то $t \leftarrow -t$.

6 Если $n \equiv 3 \pmod 4$ и $r \equiv 3 \pmod 4$, то $t \leftarrow -t$.

7 Установить $u \leftarrow n \bmod r$.

8 Установить $n \leftarrow r$.

9 Если $n > 1$, то перейти к шагу 2.

10 Возвратить t.

Библиография

[1] Лидл Р., Нидеррайтер Г. Конечные поля
М.: Мир, 1988

[2] American National Standard X9.62-2005. Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)
American National Standards Institute, 2005

[3] ISO/IEC 10646:2012 Information technology – Universal Coded Character Set (UCS)
International Organization for Standardization, 2012

[4] PKCS #5: Password-Based Encryption Standard. Version 2.0
RSA Laboratories, 1999

[5] Shamir A. Identity-based cryptosystems and signature schemes
In: Advances in Cryptology — CRYPTO'84 Proceedings, Lecture Notes in Computer Science, v. 196: 47–53, Springer Verlag, 1985

[6] Schnorr C. P. Efficient Signature Generation by Smart Cards
J. Cryptology, 4(3): 161–174, 1991

[7] Hankerson D., Menezes A., Vanstone S. Guide to Elliptic Curve Cryptography
N. Y.: Springer, 2004