Электронная цифровая подпись (ЭЦП). Хэш-функции

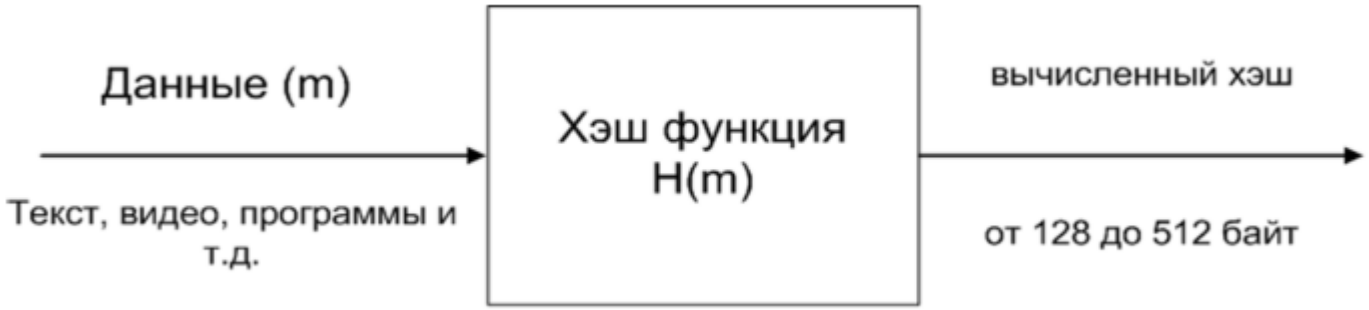
Лекция №8

Понятие хэш-функции

Хэш-функции - это функции, математические или иные, которые получают на вход строку произвольной длины (прообраз) и преобразуют ее в значение фиксированной, обычно меньшей длины **h**

$$h=H(m)$$

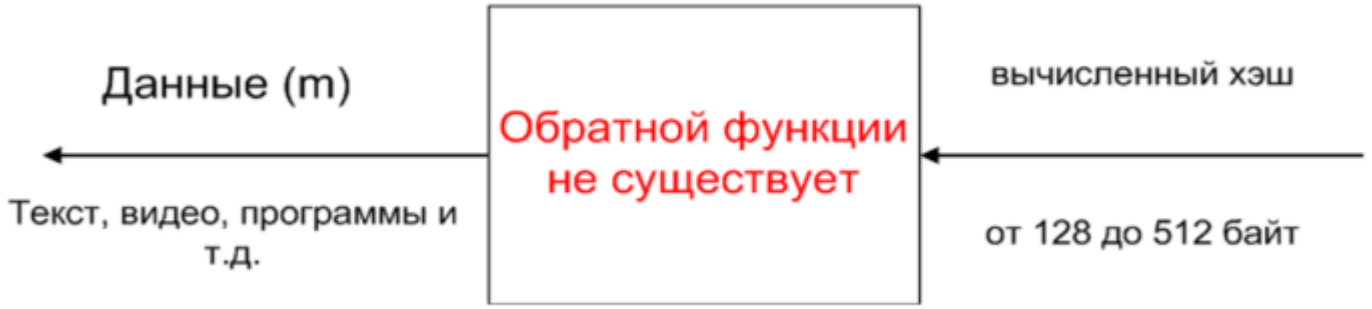
где **h** имеет фиксированную длину



Требования к хэш-функции

Хэш-функция должна обладать следующими свойствами:

- **Простота:** зная **M** легко вычислить **h**
- **Однонаправленность**: если мы знаем значение хэш-функции **h**, то задача нахождения сообщения **M** такого, что **H(M) = h**, должна быть вычислительно трудной



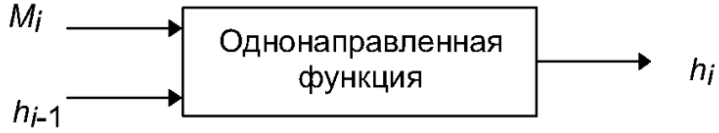
- **Устойчивость к столкновениям**: при заданном сообщении **M** задача нахождения другого сообщения **M'**, такого, что **H(M) = H(M')**, должна быть вычислительно трудной

Свойства хэш-функции

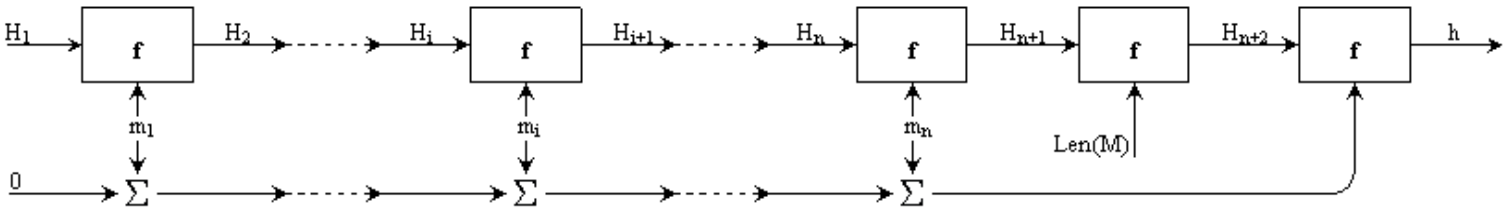
- Хэш функция является **открытой**, тайны ее расчета не существует
- Безопасность однонаправленной хэш-функции заключается именно в ее **однонаправленности**
- У выхода **нет видимой зависимости** от входа
- Изменение **одного бита** прообраза приводит к изменению, в среднем, **половины битов** значения хэш-функции
- Вычислительно **невозможно найти прообраз**, соответствующий заданному значению хэш-функции

Свойства хэш-функции

- Однонаправленные хэш-функции строятся на идее **сжатия информации**



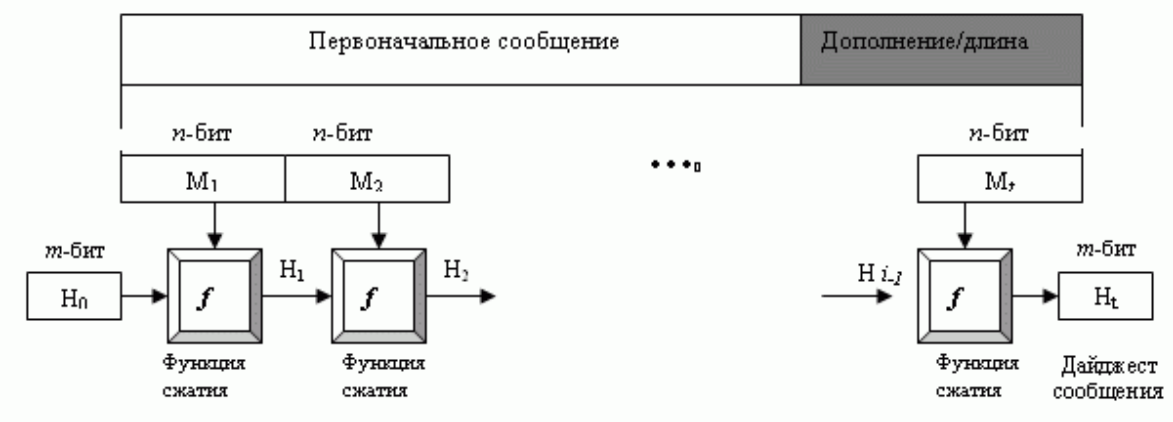
- Входами функции сжатия являются блок сообщения и значение хэш-функции предыдущего блока текста
- Выход представляет собой хэш-значение всех блоков до этого момента
- Хэш-значением всего сообщения является хэш-значение последнего блока



Хэш на базе симметричных шифров

- Итеративная функция криптографического хэширования может использовать блочный шифр с симметричными ключами как функцию сжатия
- Вместо хэш-функции с вводом переменного размера создана и используется необходимое количество раз функция с вводом фиксированного размера, называемая **функцией сжатия**
- Функция сжатия сжимает **n**-битовую строку и создает **m**-битовую строку, где **n** обычно больше чем **m**. Эта схема известна как **итеративная криптографическая функция**
- Наиболее известными схемами построения хэш-функций являются:
  - Схема Меркеля-Дамгарда (Merkle-Damgard)
  - Схема Рабина
  - Схема Девиса-Мейера
  - Схема Матиса-Мейера-Осеаса
  - Схема Миагучи-Пренеля

Схема Меркеля-Дамгарда



Описание алгоритма:

- Длина сообщения и дополнение добавляются в конец сообщения, чтобы создать увеличенное сообщение, которое может быть равномерно разделено на n -битовые блоки; здесь n - размер блока, который будет обработан функцией сжатия
- Сообщение рассматривают как t блоков, размер каждого состоит из n бит. Мы обозначим каждый блок M_1,..., M_t. Мы обозначаем дайджест, созданный при t итерациях, - H_1, H_2,...., H_t
- Перед стартом итерации дайджест H_0 устанавливается на фиксированное значение, обычно называемое IV (начальное значение или начальный вектор)
- Функция сжатия при каждой итерации обрабатывает H_{i-1} и M_i, создавая новый H_i. Другими словами, мы имеем $H_i = f(H_{i-1}, M_i)$, где f - функция сжатия
- H_t - функция криптографического хэширования первоначального сообщения, то есть h(M)

Схема Рабина

- Итеративная хэш-функция, предложенная Рабиным, очень проста
- Схема Рабина базируется на схеме Меркеля-Дамгарда
- Функция сжатия заменяется любым алгоритмом шифрования
- Блок сообщения используется как ключ
- Предварительно созданный дайджест используется как исходный текст
- Зашифрованный текст - новый дайджест сообщения

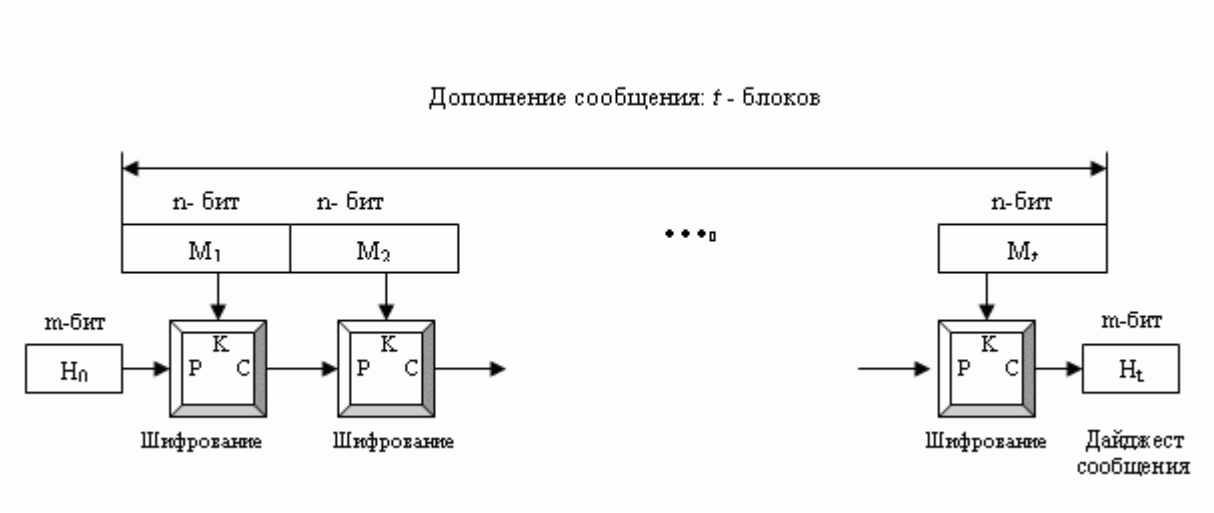


Схема Дэвиса-Мейера

- Схема Девиса-Мейера (Davies-Mayer). В основном она повторяет схему Рабина, за исключением того, что использует дополнительную функцию сложения по модулю 2 с предыдущим блоком

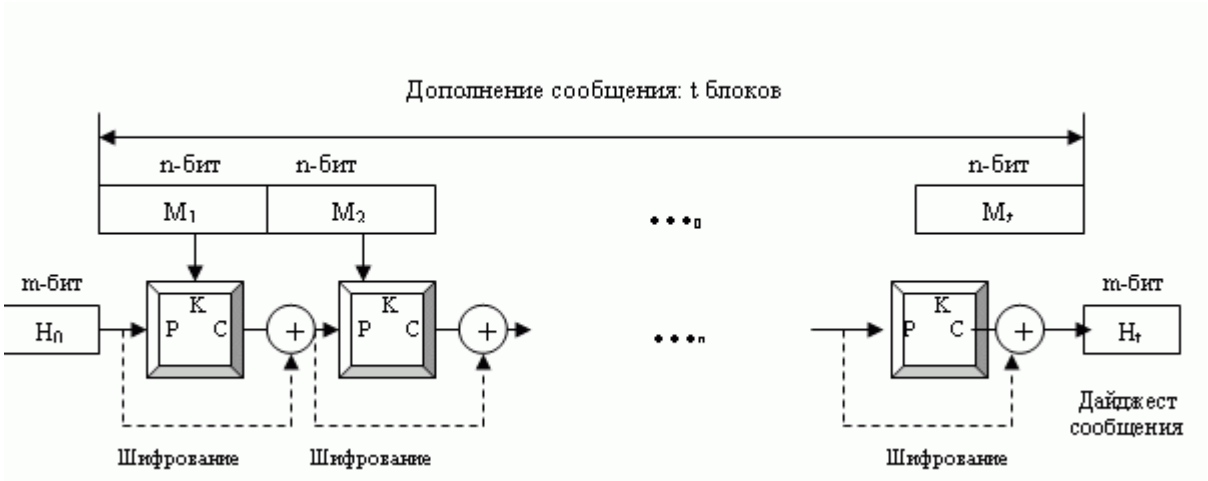


Схема Матиса-Мейера-Осеаса

- Это версия схемы Девиса-Мейера: блоки сообщения применяются как ключи криптосистемы. Схема может быть использована, если блоки данных и ключ шифрования имеют один и тот же размер. Например, AES хорошо подходит для этой цели

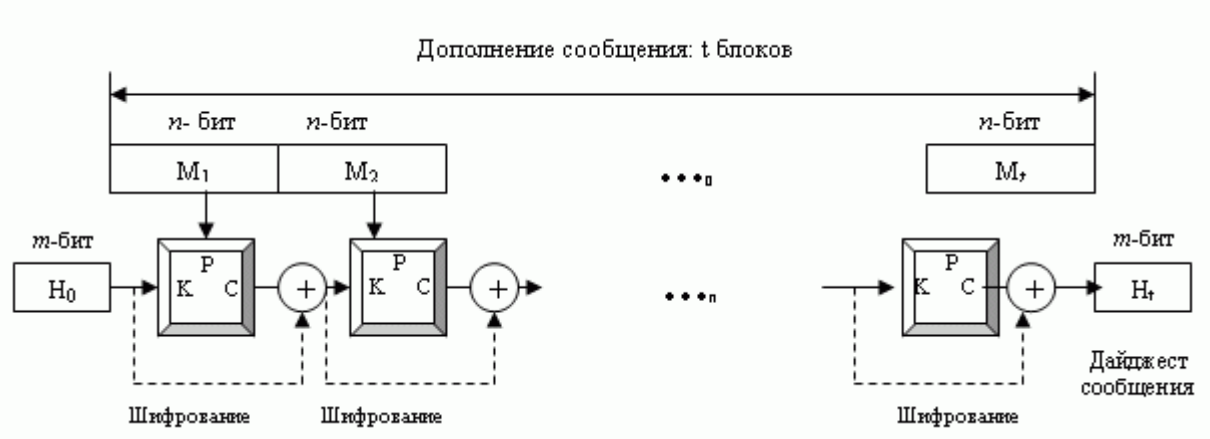
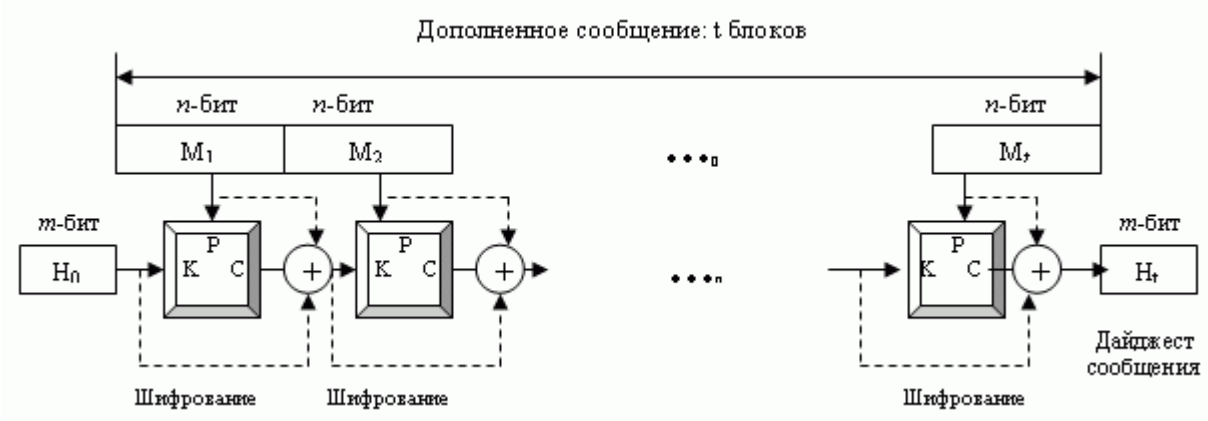


Схема Миагучи-Пренеля

- Схема Миагучи-Пренеля - расширенная версия схемы Матиса-Мейера-Осеаса. Чтобы сделать алгоритм более устойчивым к атаке, исходный текст, ключ шифра и зашифрованный текст складываются с помощью ИСКЛЮЧАЮЩЕГО ИЛИ и создают новый дайджест



Области применения хэш-функций

- **Проверка целостности данных** - обнаружение изменений в исходном тексте

  Сохранение хэш-кода и последующее сравнение с эталоном повторно вычисленного для тех же данных хэш-значения. Неравенство величин означает нарушение целостности

- **Системы аутентификации**

  Например, хэширование паролей позволяет хранить не сам пароль, а только его hash, что для проверки доступа вполне достаточно

- **Электронная цифровая подпись (ЭЦП)**

  Поскольку подписываемые документы — переменного (и как правило достаточно большого) объёма, в схемах **ЭЦП** зачастую подпись ставится не на сам документ, а на его **хэш**. Для вычисления хэша используются криптографические хэш-функции, что гарантирует выявление изменений документа при проверке подписи. Хэш-функции не являются частью алгоритма **ЭЦП**, поэтому в схеме может быть использована любая надёжная хэш-функция

Терминология

Значение хэш-функции также называют:

- Хэш-кодом
- Функцией (значением) свертки
- Профилем сообщения
- Дайджестом сообщения
- Криптографической контрольной суммой
- Цифровым отпечатком
- Кодом аутентичности сообщения
- Кодом обнаружения манипуляций

Известные хэш-функции

- Функции семейства **MD** (Рональд Ривест):
  - **MD2** - Message Digest #2: Низкоскоростной, но очень надежный алгоритм, создающий 128-разрядные дайджесты данных любого объема
  - **MD4** - Message Digest #4 (1990): Более скоростной, но менее надежный алгоритм, создающий 128-разрядные дайджесты данных любого объема. 512-битовые блоки. Есть дефекты.
  - **MD5** Message Digest #5 (1992): Версия MD4 с повышенной надежностью, преимущества также и в скорости. 128-разрядные дайджесты данных любого объема
- Функции семейства **SHA**:
  - **SHA -** Secure Hash Algorithm (1992): 160-разрядные хэш-код (дайджест), не устойчив к коллизиям, использует 512-битовые блоки
  - **SHA-1** - Secure Hash Algorithm 1 (1995): Модификация SHA: Исправлены недостатки, Решает проблему коллизий
- **MAC** - Message Authentication Code
  - При создании хэша (дайджеста) используется также секретный ключ, использует 128-битную хэш-функцию

Функция хеширования MD-2

- **MD2** - это 128-битовая однонаправленная хэш-функция, разработанная Роном Ривестом
- Безопасность MD2 опирается на случайную **перестановку байтов**. Эта перестановка фиксирована и зависит от разрядов
- **Алгоритм работы**

1.Дополните сообщение i байтами, значение i должно быть таким, чтобы длина полученного сообщения была кратна 16 байтам.

2.Добавьте к сообщению 16 байтов контрольной суммы

3.Проинициализируйте 48-байтовый блок: X0, X1 ,X2 ,… , X47. Заполните первые 16 байтов X нулями, во вторые 16 байтов X скопируйте первые 16 байтов сообщения, а третьи 16 байтов X должны быть равны XOR первых и вторых 16 байтов X

4.Вот как выглядит функция сжатия:

```
t=0;
For j = 0 to 17;
For k = 0 to 47;
t = Xt XOR St;
Xk=t;
t = (t +j) mod 256
```

5.Скопируйте во вторые 16 байтов X вторые 16 байтов сообщения, а третьи 16 байтов X должны быть равны XOR первых и вторых 16 байтов X. Выполните этап (4). Повторяйте этапы (5) и (4) по очереди для каждых 16 байтов сообщения

6.Выходом являются первые 16 байтов X

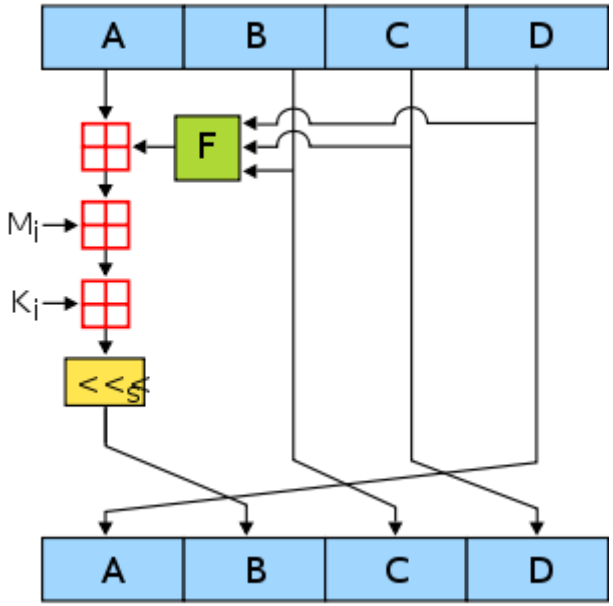
Функция хеширования MD-4

Цели разработки нового алгоритма

- **Безопасность**. Вычислительно невозможно найти два сообщения с одинаковым хэш-значением. Вскрытие грубой силой является самым эффективным
- **Прямая безопасность**. Безопасность MD4 не основывается на каких-либо допущениях, например, предположении о трудности разложения на множители
- **Скорость**. MD4 подходит для высокоскоростных программных реализаций. Она основана на простом наборе битовых манипуляций с 32-битовыми операндами
- **Простота и компактность**. MD4 проста, насколько это возможна, и не содержит больших структур данных или сложных программных модулей
- **Удачная архитектура**. MD4 оптимизирована для микропроцессорной архитектуры (особенно для микропроцессоров Intel), для более крупных и быстрых компьютеров можно выполнить любые необходимые изменения

Функция хеширования MD-4

- **MD4** - это 128-битовая однонаправленная хэш-функция, разработанная Роном Ривестом в 1990 году, как модификация MD-2

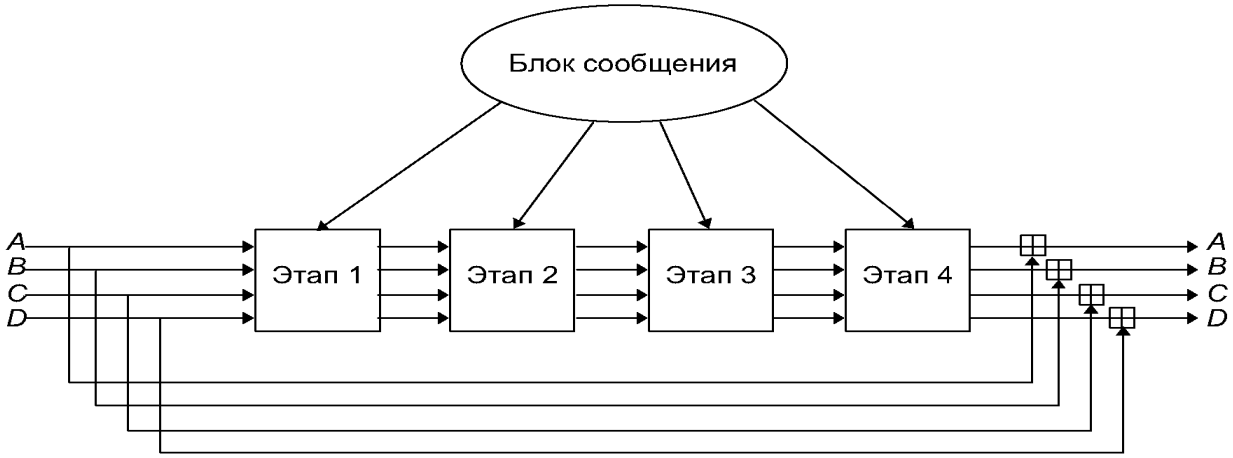


Одна операция MD4. Хеширование с MD4 состоит из 48 таких операций, сгруппированных в 3 раунда по 16 операций. F — нелинейная функция; в каждом раунде функция меняется. Mi означает 32-битный блок входного сообщения, а Ki — 32-битная константа, различная для каждой операции.

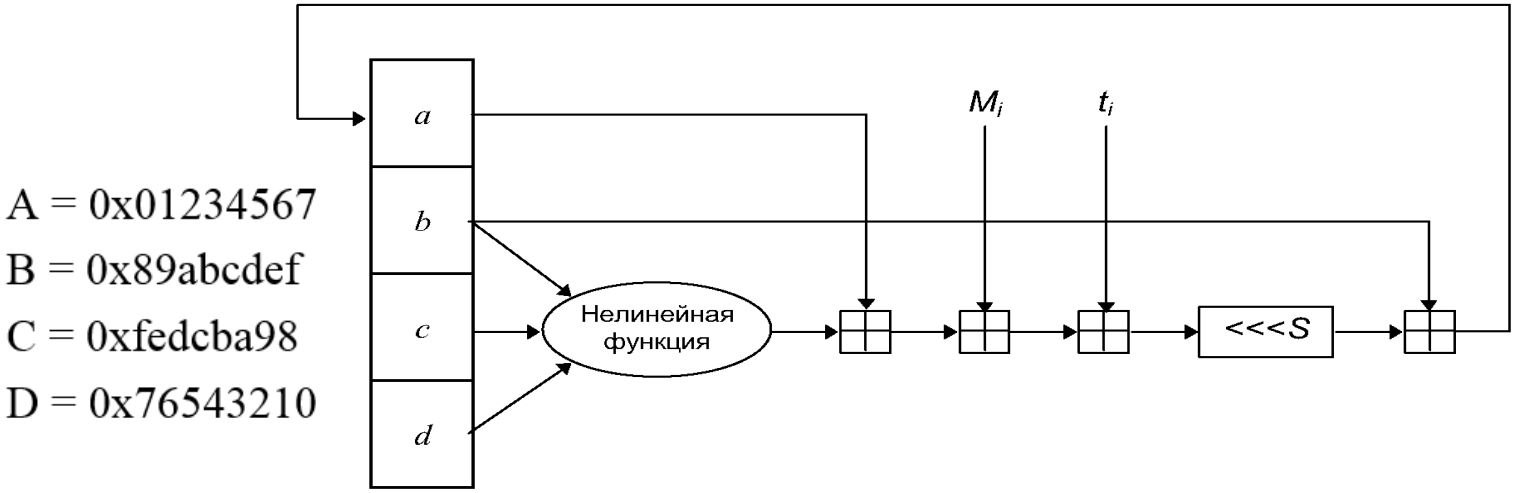
- Подробнее - https://ru.wikipedia.org/wiki/MD4

Функция хеширования MD-5

- **MD5** – обрабатывает исходный текст блоками по 512 бит
- К исходной последовательности приписывается единица и необходимое количество нулей, чтобы ее длина стала на 64 бита меньше числа, кратного 512.
- Затем приписывается 64-битное представление длины исходной последовательности
- Главный цикл алгоритма состоит из 4-х этапов по 16 операций в каждом этапе



Одна операция MD-5



- Каждая операция представляет собой нелинейную функцию над тремя переменными **a**, **b**, **c** и **d**
- Затем она добавляет этот результат к четвертой переменной, подблоку текста $\mathbf{M_i}$ и константе $\mathbf{t_i}$
- Далее результат циклически сдвигается вправо на переменное число битов и добавляет результат к одной из переменных **a**, **b**, **c** и **d**
- Наконец результат заменяет одну из переменных **a**, **b**, **c** и **d**

Одна операция MD-5

Нелинейные функции:

- Для первого этапа:

  $FF\ (a, b, c, d, M_j, s, t_i) \rightarrow a = b + ((a + F(b,c,d) + M_j + t_i) <<< s)$

- Для второго этапа:

  $GG\ (a, b, c, d, M_j, s, t_i) \rightarrow a = b + ((a + G(b,c,d) + M_j + t_i) <<< s)$

- Для третьего этапа:

  $RR\ (a, b, c, d, M_j, s, t_i) \rightarrow a = b + ((a + R(b,c,d) + M_j + t_i) <<< s)$

- Для четвертого этапа:

  $HH\ (a, b, c, d, M_j, s, t_i) \rightarrow a = b + ((a + H(b,c,d) + M_j + t_i) <<< s)$

Одна операция MD-5

Значения констант:

FF (a, b, c, d, M_0, 7, 0xd76aa478)	FF (a, b, c, d, M_8, 7, 0x698098d8)
FF (d, a, b, c, M_1, 12, 0xe8c7b756)	FF (d, a, b, c, M_9, 12, 0x8b44f7af)
FF (c, d, a, b, M_2, 17, 0x242070db)	FF (c, d, a, b, M_{10}, 17, 0xffffSbbl)
FF (b, c, d, a, M_3, 22, 0xclbdceee)	FF (b, c, d, a, M_{11}, 22, 0x895cd7be)
FF (a, b, c, d, M_4, 7, 0xf57c0faf)	FF (a, b, c, d, M_{12}, 7, 0x6b901122)
FF (d, a, b, c, M_5, 12, 0x4787c62a)	FF (d, a, b, c, M_{13}, 12, 0xfd987193)
FF (c, d, a, b, M_6, 17, 0xa8304613)	FF(c, d, a, b, M_{14}, 17, 0xa679438e)
FF (b, c, d, a, M_7, 22, 0xfd469501)	FF(b, c, d, a, M_{15}, 22, 0x49b40821)

Константы, t_i являются целой частью 232*abs(sin(i)),
где i – номер операции

Подробнее - https://ru.wikipedia.org/wiki/MD5

Функция хэширования SHA–1

Основное отличие алгоритма SHA от других защищенных функций хэширования (MD2, MD4, MD5) – генерирование 160-битового хэш-кода (против 128-битового)

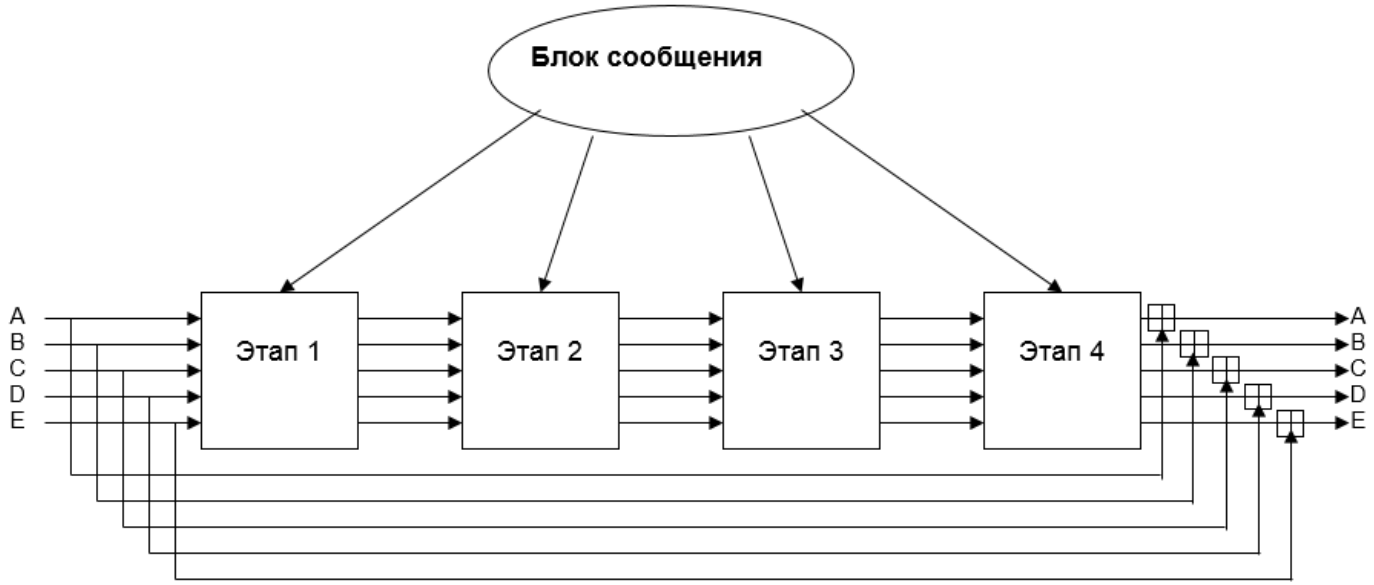
> Современные технологии распределенных вычислений и многопроцессорные компьютеры демонстрируют недостаточную защищенность 128-битовых хэш-кодов. «Кроме того, были разработаны сценарии целого ряда атак, демонстрирующих уязвимость MD5 в отношении современных методов криптоанализа»

- Задача поиска двух наборов входных данных с одинаковым значением хэш-функции имеет сложность $O(2^{80})$
- Задача нахождения набора входных данных с заданным значением имеет сложность $O(2^{160})$

Основные характеристики SHA-1

- Длина хэш-кода - 160 бит
- Длина обрабатываемых блоков - 512 бит
- Число шагов алгоритма - 80 (4 раунда по 20 шагов)
- Максимальная длина хэшируемых данных - $2^{64}-1$
- Число базовых функций - 4
- Число аддитивных констант - 4

Главный цикл SHA–1



Главный цикл состоит из четырех этапов по 20 операций в каждом

Одна операция SHA–1

1.На вход поступает k-битовый блок данных, где $k < 2^{64}$

2.k-битовый блок дополняется так, чтобы его длина стала кратной 512 разрядам (данные обрабатываются 512-битовыми блоками).

3.К полученному результату добавляется 64-битовое представление длины исходного блока данных

4. Инициализируются пять 32-разрядных переменных:

A = 0x67452301

B = 0xefcdab89

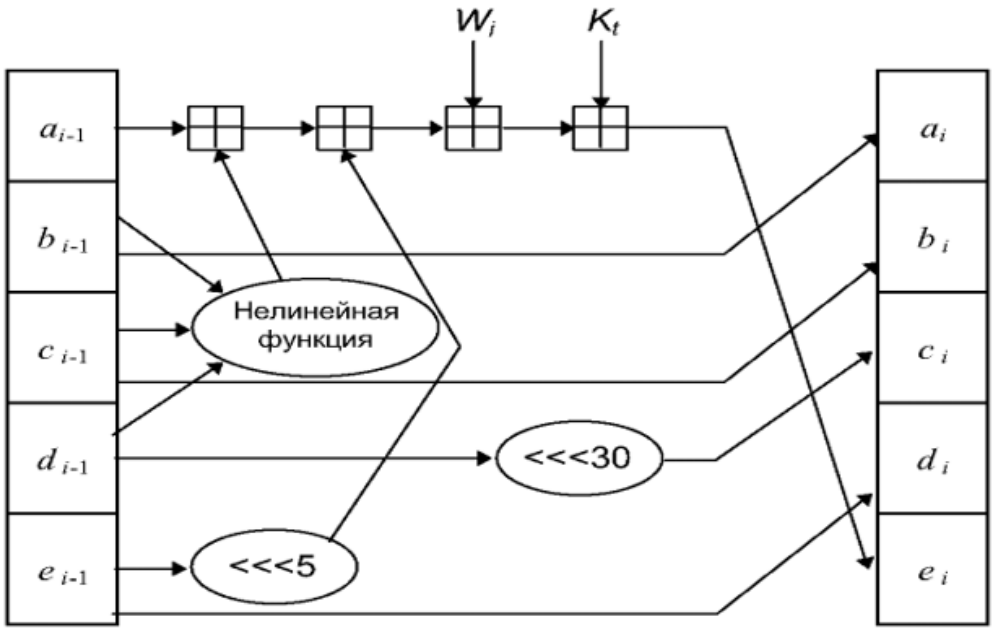
C = 0x98badcfe

D = 0x10325476

E = 0xc3d2e1f0

5.Производится обработка 512-битовых блоков данных в 4 раунда по 20 операций каждый

Одна операция SHA–1



A = 0x67452301
B = 0xefcdab89
C = 0x98badcfe
D = 0x10325476
E = 0xc3d2e1fO

```
FOR t = 0 to 79
TEMP = (a <<< 5) +F(b,c,d) + e + Wt + Kt
e = d
d=c
c = b<<<30
a = TEMP
```

SHA-1: Формирование констант

$$W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) \ll 1 \quad (16 \le t \le 79)$$

Соответствие аддитивных констант K_t и нелинейных функций F_t номеру операции
80 операций = 4 раунда по 20 шагов

Номер шага (t)	Нелинейная функция, (F_t)	Аддитивная константа (K_t)
$0 \le t \le 19$	$(X \wedge Y) \vee ((\neg X) \wedge Z)$	$0x5a827999 = 2^{30} \times 2^{1/2}$
$20 \le t \le 39$	$X \oplus Y \oplus Z$	$0x6ed9eba1 = 2^{30} \times 3^{1/2}$
$40 \le t \le 59$	$(X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$	$0x8f1bbcdc = 2^{30} \times 5^{1/2}$
$60 \le t \le 79$	$X \oplus Y \oplus Z$	$0xca62c1d6 = 2^{30} \times 10^{1/2}$

Определение ЭЦП

Электронная цифровая подпись (ЭЦП) предназначена для аутентификации лица, подписавшего электронный документ. Кроме этого, использование цифровой подписи позволяет осуществить:

- **Контроль целостности** передаваемого документа: при любом случайном или преднамеренном изменении документа подпись станет недействительной, потому что вычислена она на основании исходного состояния документа и соответствует лишь ему
- **Защиту от изменений** (подделки) документа: гарантия выявления подделки при контроле целостности делает подделывание нецелесообразным в большинстве случаев
- **Невозможность отказа от авторства**. Так как создать корректную подпись можно, лишь зная закрытый ключ, а он должен быть известен только владельцу, то владелец не может отказаться от своей подписи под документом
- **Доказательное подтверждение авторства документа**: Так как создать корректную подпись можно, лишь зная закрытый ключ, а он должен быть известен только владельцу, то владелец пары ключей может доказать своё авторство подписи под документом. В зависимости от деталей определения документа могут быть подписаны такие поля, как «автор», «внесённые изменения», «метка времени» и т. д.

Из чего состоит ЭЦП?

Для работы с ЭЦП необходимы:

- **Битовая строка**, присоединяемая к документу после подписания (которая и называется электронной цифровой подписью)
- **Протокол**, с помощью которого получатель убеждается в подлинности отправителя и целостности сообщения, называется проверкой подлинности (аутентификацией)

ЭЦП строится на основе двух компонент:

- Содержания информации, которая подписывается
- Личной информации (код, пароль, ключ) того, кто подписывает

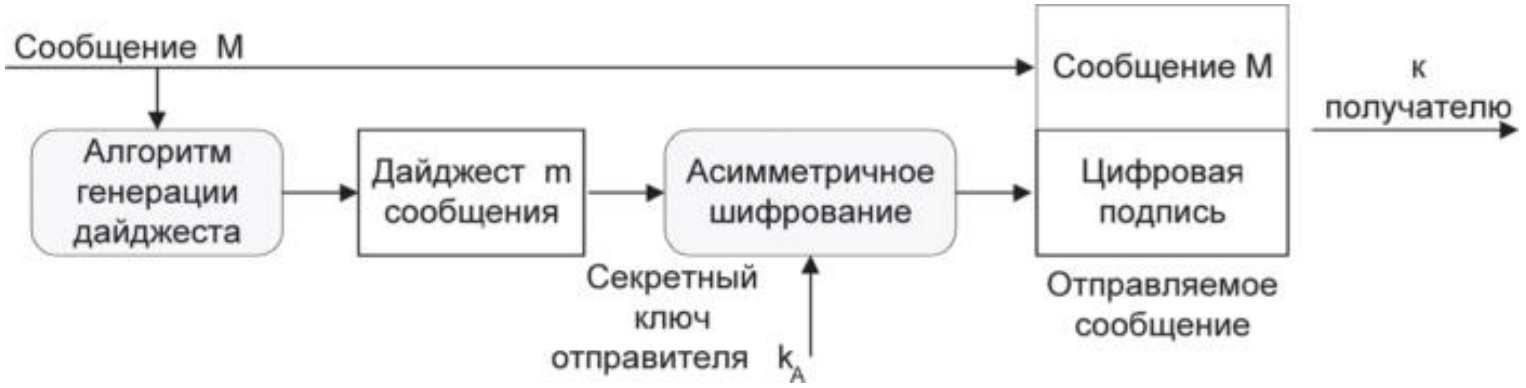
Алгоритмы ЭЦП

Существует несколько схем построения цифровой подписи

- На основе алгоритмов **симметричного шифрования**. Данная схема предусматривает наличие в системе третьего лица — арбитра, пользующегося доверием обеих сторон. Авторизацией документа является сам факт зашифрования его секретным ключом и передача его арбитру
- На основе алгоритмов **асимметричного шифрования**. На данный момент такие схемы ЭЦП наиболее распространены и находят широкое применение
- Кроме этого, существуют другие разновидности цифровых подписей (групповая подпись, неоспоримая подпись, доверенная подпись), которые являются модификациями описанных выше схем. Их появление обусловлено разнообразием задач, решаемых с помощью ЭЦП

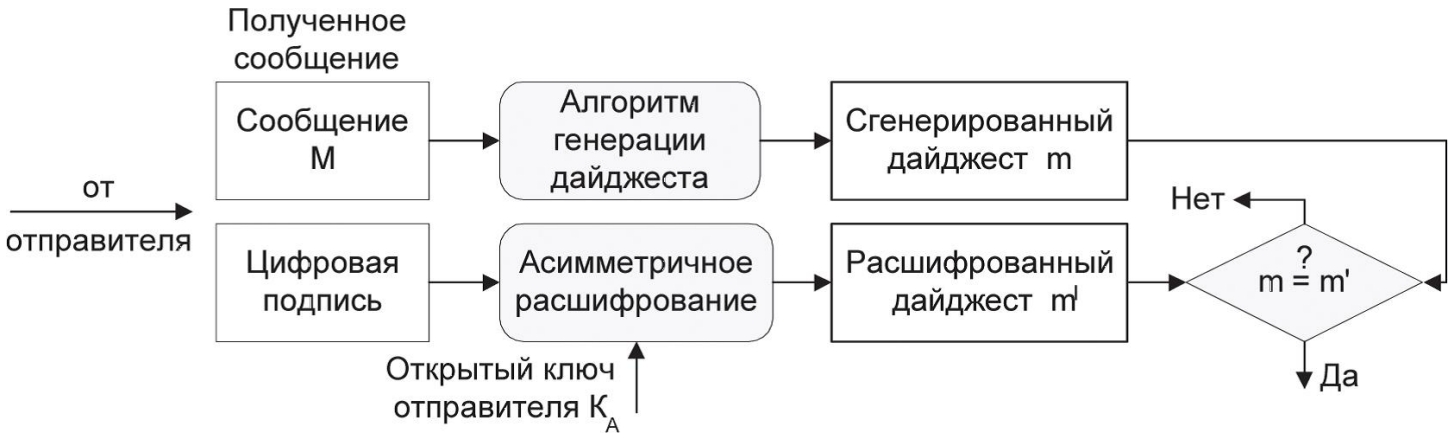
Формирование ЭЦП

- На подготовительном этапе этой процедуры абонент А — отправитель сообщения — генерирует пару ключей: секретный ключ $\mathbf{k_A}$ и открытый ключ $\mathbf{К_A}$
- Для формирования цифровой подписи отправитель А прежде всего вычисляет значение хэш- функции **h(M)** подписываемого текста **M**



Проверка ЭЦП

- При проверке ЭЦП абонент В — получатель сообщения **M** — расшифровывает принятый дайджест **m** открытым ключом $К_А$ отправителя А. Кроме того, получатель сам вычисляет с помощью хэш-функции **h(M)** дайджест **m** принятого сообщения **M** и сравнивает его с расшифрованным
- Если эти два дайджеста **m** и **m'** совпадают, то цифровая подпись является подлинной



Наиболее известные схемы ЭЦП

- DSA
- RSA
- Эль-Гамаля (ElGamal)
- Рабина
- Шнорра
- Диффи-Лампорта

ЭЦП с помощью RSA

Формирование цифровой подписи:

1. Отправитель сжимает сообщение **M** при помощи криптографической хеш-функции **h** в целое число **m =h(M)**
2. Отправитель вычисляет значение цифровой подписи **S** для сообщения **M** на основе ранее полученного значения хеш-образа **m** и значения своего закрытого (секретного) ключа подписи $\mathbf{K_C}$. Для этого используется преобразование, аналогичное преобразованию, выполняемому при шифровании по алгоритму RSA:

$$S = m^{K_C} \bmod r$$

3. Пара **(M,S)**, представляющая собой подписанное отправителем сообщение, передаётся получателю. Сформировать подпись **S** мог только обладатель закрытого ключа $\mathbf{K_C}$

ЭЦП с помощью RSA

Проверка подлинности сообщения

1.Получатель сжимает полученное сообщение **M’** при помощи криптографической хеш-функции **h**, идентичной той, которая была использована отправителем, в целое число **m’**.

2. Получатель выполняет расшифрование открытым ключом $\mathbf{K_o}$ отправителя дайджеста **m** оригинального сообщения, преобразуя значение подписи **S** по алгоритму RSA:

$$m = S^{K_o} \bmod r$$

3. Получатель сравнивает полученные значения **m’** и **m**. Если данные значения совпадают, т. е.

$$S^{K_o} \bmod r = h(M')$$

то получатель признает полученное сообщение подлинным и принадлежащим отправителю

ЭЦП Эль-Гамаля

Формирование подписи

1. Для того чтобы подписать сообщение **M**, сначала отправитель хэширует его с помощью хэш-функции **h(M)** в целое число **m**:

   $$m = h(M),\ 1<m<(P-1),$$

2. Генерирует случайное целое число **K**, $1 < K < (P-1)$, такое, что **K** и **(P-1)** являются взаимно простыми. Затем отправитель вычисляет целое число **a**:

   $$a = G^K \bmod P$$

3. Применяя расширенный алгоритм Евклида, вычисляет с помощью секретного ключа **X** целое число **b** из уравнения

   $$m = X * a + K * b \pmod{(P-1)}$$

4. Пара чисел **(a,b)** образует цифровую подпись **S**:

   S = (a, b), проставляемую под документом **M**

5. Тройка чисел **(M, a, b)** передается получателю, в то время как пара чисел **(X, K)** держится в секрете

ЭЦП Эль-Гамаля

Проверка подписи

1. После приема подписанного сообщения **(M, a, b)** получатель должен проверить, соответствует ли подпись **S = (a, b)** сообщению **M**
2. Для этого получатель сначала вычисляет по принятому сообщению **M** число

   m = h(M), т.е. хэширует принятое сообщение **M**

3. Затем получатель вычисляет значение

   $$A = Y^a a^b \pmod P$$

4. Признает сообщение M подлинным, если, и только если

   $$A = G^m \pmod P$$

   Иначе говоря, получатель проверяет справедливость соотношения:

   $$Y^a a^b \pmod P = G^m \pmod P$$

DSA – алгоритм цифровых сигнатур

- Алгоритм с открытым ключом
- Размер ключа – от 512 до 1024 бит
- При проверке достоверности сигнатуры DSA работает в 10-40 раз медленнее RSA
- Используется только для ЭЦП, не для шифрования
- DSS – стандарт США (1994) на основе DSA и алгоритме хэширования SHA-1

Алгоритм DSA: Параметры

Открытый ключ (p, q, g, y):

- **p** – простое число длиной от 512 до 1024 битов
- **q** – 160-битовый простой множитель p-1
- **g** = $h^{(p-1)/q} \bmod p$, где h – любое число < (p-1), для которого $h^{(p-1)/q} \bmod p > 1$
- **y** = $g^x \bmod p$ (p-битовое число)

Закрытый ключ (x):

- X < q (160-битовое число)

Алгоритм DSA

Подпись сообщения:

1. Алиса генерирует случайное число **k**, меньшее **q**
2. Алиса генерирует:

$$r = (g^k \bmod p) \bmod q$$

$$s = (k^{-1} (H(m) + x\, r)) \bmod q$$

3. Алиса посылает Бобу свою подпись - **(r, s)**

Проверка подписи:

1. Боб вычисляет:

$$w = s^{-1} \bmod q$$

$$u1 = (H(m) * w) \bmod q$$

$$u2 = (r * w) \bmod q$$

$$v = ((g^{u1} * y^{u2}) \bmod p) \bmod q$$

2. Если **v = r** , то подпись верна

Национальные стандарты ЭЦП

- На базе вычисления **логарифма в конечном поле**:
  - Стандарт США - **Digital Signature Standard** (принят в 1991 г. с последующими изменениями в 1993, 1996г.)
  - Российский стандарт цифровой подписи **ГОСТ Р 34.10-94**
  - Стандарт Республики Беларусь **СТБ 1176.2** (1999г.)
- На **технике эллиптических кривых**:
  - Международная серия стандартов **OSO/IEC 14946**
  - Стандарт **IEEE P1363**
  - Новый стандарт РФ – **ГОСТ Р 34.10-2001**

Распределение ключей ЭЦП

- В настоящее время реализованы и опубликованы схемы **механизмов взлома ЭЦП**, основанные на генерации новой пары (открытый, секретный) ключей и включении нового открытого ключа в конверт ЭЦП

Распределение ключей осуществляется двумя способами:

- созданием **центра генерации и распределения ключей** - в этом случае компрометация центра приводит к компрометации всей передаваемой информации
- **прямым обменом ключами** между абонентами - здесь необходимо обеспечить подлинность каждого абонента

Защита ключей

- Принципиальным моментом в системе ЭЦП является невозможность подделки ЭЦП пользователя без знания его секретного ключа подписывания. Поэтому необходимо защитить секретный ключ подписывания от несанкционированного доступа. Секретный ключ ЭЦП, аналогично ключу симметричного шифрования, рекомендуется хранить на персональном ключевом носителе в защищенном виде

USB-брелок

Таблетки Touch-Memory

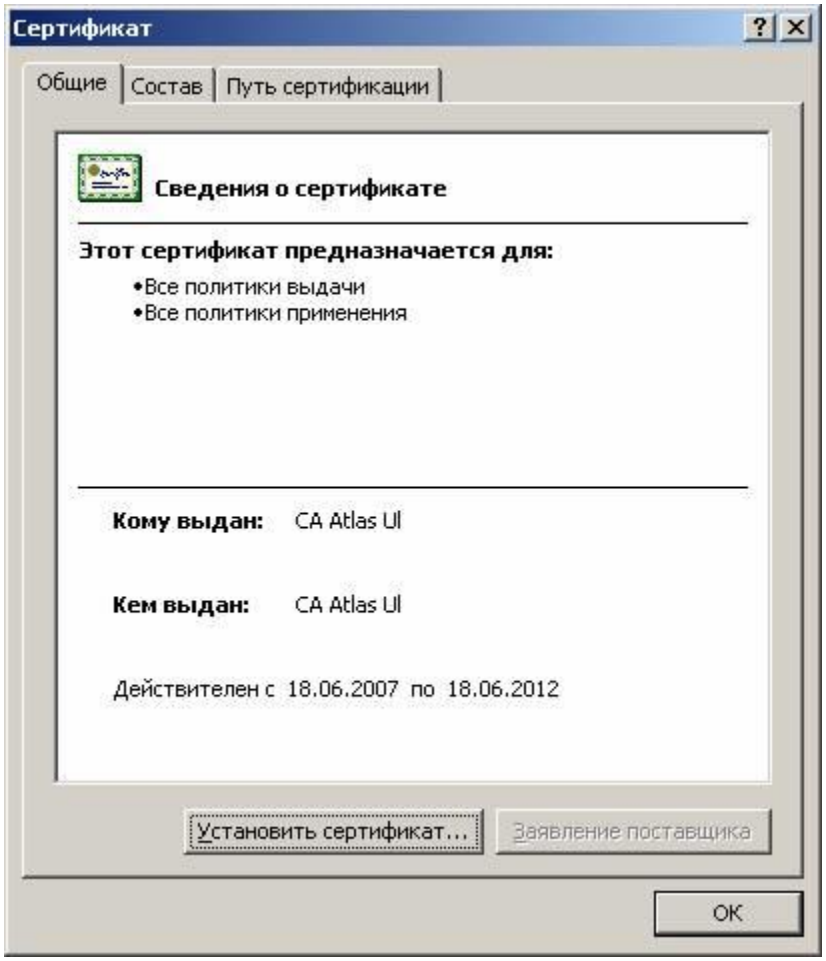
Понятие сертификата

- **Сертификат электронной подписи** — это бумажный или электронный документ, позволяющий проверить подлинность электронной подписи, он подтверждает принадлежность ключа проверки электронной подписи владельцу сертификата
- Сертификат ЭЦП выпускается **удостоверяющим центром**, который занимается выдачей и обслуживанием сертификатов электронной подписи, или его доверенным лицом.
- Термин «сертификат ключа проверки» или «открытый ключ» подразумевает, что этот документ содержит все данные о владельце закрытого ключа электронной подписи, в том числе и данные по удостоверяющему центру, который выпустил подпись. Удостоверяющий центр, выдавший сертификат, отвечает за достоверность данных, которые содержатся в нём

Содержание сертификата

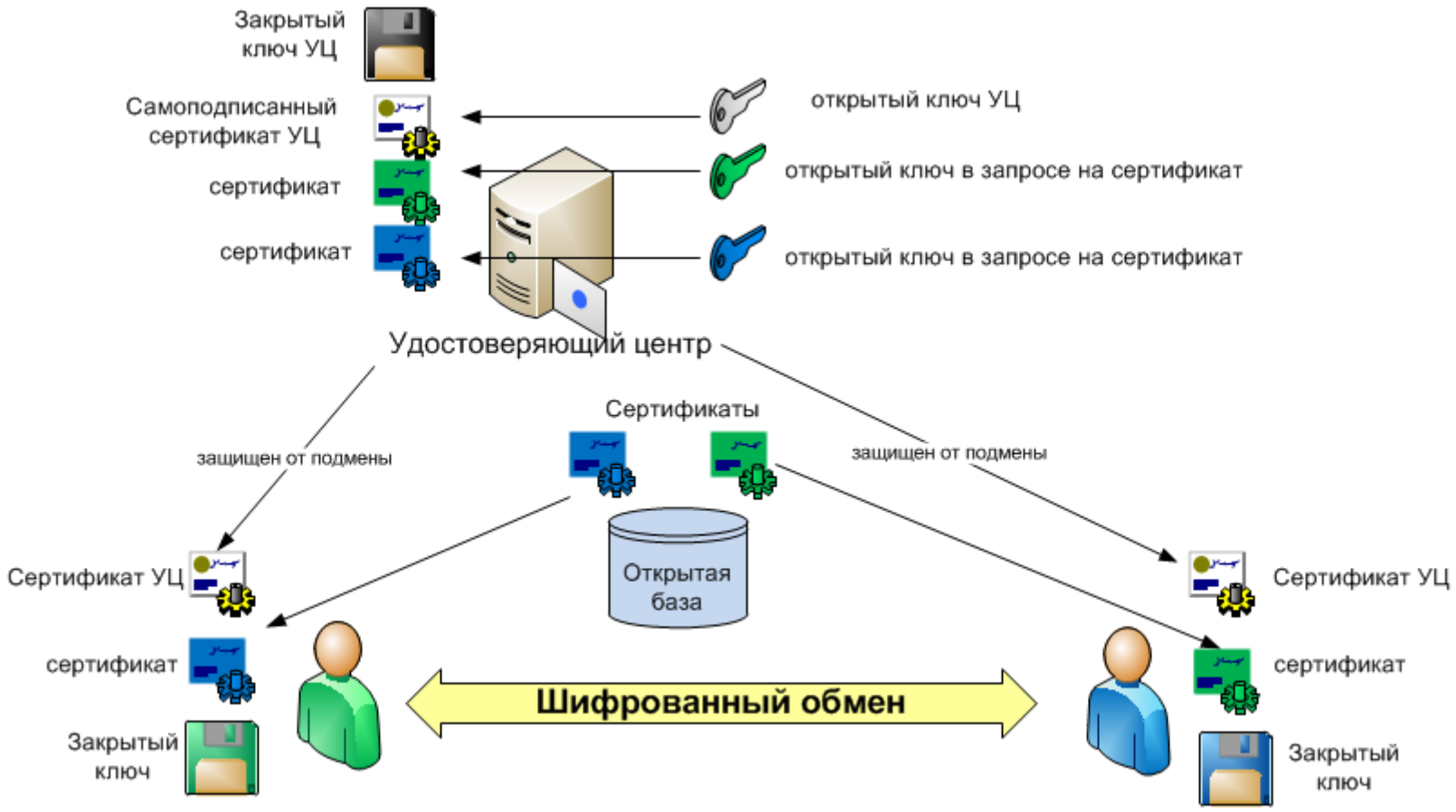
- Ключ проверки электронной подписи
- Наименование средства электронной подписи
- Наименование удостоверяющего центра, выпустившего сертификат
- Сведения об ограничениях в сфере применения ЭЦП
- Дата начала и окончания срока действия сертификата
- Сведения о владельце электронной подписи, включая:
  - ФИО и СНИЛС — для физического лица
  - ИНН физического или юридического лица, в зависимости от типа заявителя
  - Наименование и местонахождение организации для владельца — юридического лица
  - Другие сведения
- Информация о точке распространения списка отозванных сертификатов

Внешний вид сертификата



ЭЦП представляет собой совокупность закрытого ключа - контейнера, обладателем которого может быть только владелец сертификата, и однозначно соответствующего этому закрытому ключу открытого ключа – сертификата. Сертификат представим в виде файла формата X.509

Защита ключей с помощью УЦ



Использование ЭЦП в России

- В России юридически значимый сертификат электронной подписи выдаёт **удостоверяющий центр**
- Правовые условия использования электронной цифровой подписи в электронных документах регламентирует федеральный закон от 10 января 2002 г. № 1-ФЗ «Об электронной цифровой подписи»
- В России в электронном документообороте можно использовать три вида подписи:
  - Простую - **ПЭП**
  - Усиленную неквалифицированную - **НЭП**
  - усиленную квалифицированную - **КЭП**

Простая ЭП (ПЭП)

- **ПЭП** — это знакомые всем коды доступа из СМС, коды на скретч-картах, пары “логин-пароль” в личных кабинетах на сайтах и в электронной почте
- Простая подпись создается средствами информационной системы, в которой ее используют, и подтверждает, что электронную подпись создал конкретный человек

Где используется?

- Простая электронная подпись чаще всего применяется при **банковских операциях**, а также для **аутентификации в информационных системах**, для **получения госуслуг**, для заверения документов внутри корпоративного электронного документооборота
- Простую электронную подпись **нельзя** использовать при подписании электронных документов или в информационной системе, которые **содержат гостайну**

Неквалифицированная ЭП (НЭП)

- **Неквалифицированная электронная подпись** создается с помощью программ криптошифрования с использованием закрытого ключа электронной подписи. НЭП идентифицирует личность владельца, а также позволяет проверить, вносили ли в файл изменения после его отправки

Где используется?

- НЭП необходима, чтобы участвовать в **электронных госзакупках** по 44-ФЗ в качестве поставщика на шести федеральных электронных торговых площадках госзакупок. Этот же вид подписи можно использовать для **внутреннего и внешнего ЭДО**, если стороны предварительно договорились об этом

Квалифицированная ЭП (КЭП)

- **Квалифицированная электронная подпись** — самый регламентированный государством вид подписи
- КЭП обязательно имеет квалифицированный сертификат в бумажном или электронном виде, структура которого определена приказом ФСБ России № 795 от 27.12.2011.
- Программное обеспечение для работы с КЭП сертифицировано ФСБ России

Где используется?

- КЭП нужна, чтобы **сдавать отчетность** в контролирующие органы, участвовать в качестве поставщика и заказчика в **электронных торгах**, работать с государственными информационными системами, обмениваться формализованными документами с ФНС, **вести документооборот** внутри компании или с ее внешними контрагентами