Ассиметричные криптосистемы

Лекция

Ревизия: 0.1

Последняя модификация: Vlad Kovtun

Дата последней модификации: 31.01.2011 21:24:00

История изменений

15.10.2009 – Версия 0.1. Первичный документ. Ковтун В.Ю.

Содержание

Лекция 5. Ассиметричные криптосистемы. Часть 1

Вопросы

1. Ассиметричные криптосистемы. Введение.
2. Криптосистема RSA.
3. Криптосистема Эль-Гамаля.
4. Хеш-функции. Современные хеш-функции.

Ассиметричные криптосистемы. Введение

От истоков криптографии до самых современных времен, криптосистемы строились на основе элементарных преобразований: подстановки и перестановки. Ручной труд, на протяжении тысяч лет, был сменен механическими, а далее и электромеханическими шифровальными и дешифровальными машинами, которые открыли новую эру в области защиты информации. Дальнейшее изобретение компьютеров, послужило новым толчком, развитию средств не только шифрования, но и криптоанализа. Одним из таких достижений, является алгоритм LUCIFER компании IBM, который был положен в основу всем хорошо известного алгоритма DES. Однако, основу всех алгоритмов продолжали составлять, все те же, подстановки и перестановки, которые производились как отправителем, так и получателем. Другими словами, отправитель и получатель обязаны обладать одним и тем же ключом, отсюда вытекает одно из ограничений симметричных криптосистем – распространение (распределение) ключей.

Указанный недостаток послужил толчком к поиску подходов к построению криптосистем способных исключить защищенный канал передачи ключей и обеспечивая защиту передаваемых сообщений по незащищенному каналу, без дополнительных преобразований. На рис. 1, показан процесс двустороннего обмена между отправителем и получателем (абонентами), при этом злоумышленник выполняет пассивную роль слушателя. В отличие от обычных систем (с секретным ключом), системы, допускающие открытую передачу (открытой) части ключа по незащищенному каналу связи, называют системами с открытым ключом. В таких системах открытый ключ (ключ шифрования), отличается от личного ключа (ключа расшифровывания), поэтому их иногда называют ассиметричными системами или двуключевыми системами.

Понятие криптосистемы с открытым ключом включает в себя такие объекты, рис. 1.

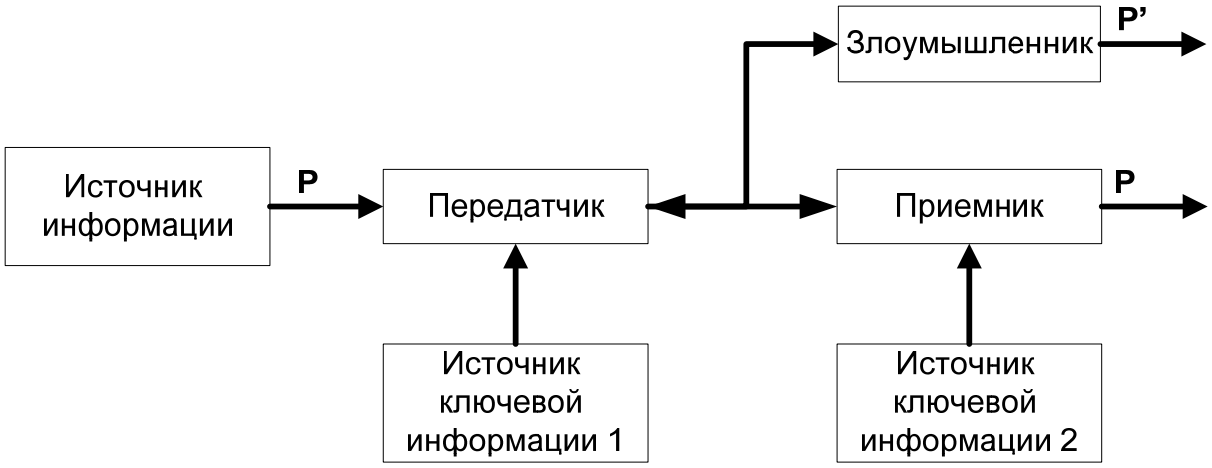

Рис. 1. Информационные потоки в криптографической системе с открытым ключом

Впервые, публично, Диффи и Хеллман в 1976 году изложили идею разработанного метода на национальной компьютерной конференции [4] и опубликована в том же году в основополагающей работе "Новые направления в криптографии" [3]. Предложеный метод решает задачу: распределения ключей и цифровой подписи. Она радикально отличалась от известных ранее подходов, за всю историю криптографии.

К числу отцов-основателей следует отнести также и Ральфа Меркля, который независимо от Диффи и Хеллмана пришел к тем же конструкциям, однако опубликовал свои результаты только в 1978 году [6].

Однако, истинную точку отсчета криптографии с открытым ключом, следует отнести к более раннему времени. Известно несколько независимых источников, которые отдают

пальму первенства, в разработке криптографии с открытым ключом, Агентству национальной безопасности США (National Security Agency - NSA).

По словам адмирала Бобби Инман (Bobby Inmann), занимал пост главы Агентства, метод криптографических преобразований с открытым ключом разработан NSA еще в середине 60-х годов. Первое документальное подтверждение этому, появилось в 1970 году в закрытом отчете Джеймса Эллиса (James Ellis) из Группы защиты электронных коммуникаций Службы безопасности Великобритании [9].

В статье энциклопедии "Британника" директор NSA Симмонс заявляет, что "двухключевая криптография была известна в Агентстве за 10 лет до публикации Диффи и Хеллмана" [7].

В опубликованной статье покойного военного криптоаналитика Джеймса Эллиса (James Ellis) из Великобритании утверждается, что ему и Клиффорду Коксу (Clifford Cocks) удалось получить работающую схему, близкую к RSA [5], за несколько лет до этого [9]. К сожалению, отсутствуют независимые подтверждения этому.

И так, в основе преобразований с открытым ключом лежит теоретико-числовой подход к определению стойкости криптоанализа, т.е. проблема обоснования стойкости криптографической схемы свелась к доказательству отсутствия полиномиального алгоритма, который решает задачу, стоящую перед злоумышленником. Из этого следует, что на данный момент стойкость криптографических схем может быть установлена лишь с привлечением каких-либо недоказуемых предположений. Поэтому, основное направление исследований состоит в поиске наиболее слабых достаточных условий для существования стойких схем каждого типа. В основном рассматриваются предположения двух типов – общие (или теоретико-сложностные) и теоретико-числовые, т.е. предположения о сложности конкретных теоретико-числовых задач [4, 3].

В работах [4, 3] показано предположение о существовании односторонних функций. Там, же дается доказательство того факта, что существование односторонних функций является необходимым и достаточным условием существования стойких криптосистем с секретным, открытым ключом и криптографических протоколов нескольких типов. В 1978 г. [2] был предложен пример такой односторонней функции $f(x)$, обладающей рядом свойств [8]:

а) существует достаточно быстрый (полиномиальный) алгоритм вычисления значений $f(x)$;

б) существует достаточно быстрый (полиномиальный) алгоритм вычисления значений обратной функции $f^{-1}(x)$;

в) функция $f(x)$ обладает некоторым «секретом», знание которого позволяет быстро вычислять значение $f^{-1}(x)$; в противном случае вычисление $f^{-1}(x)$ становится трудно разрешимой в вычислительном отношении задачей, требующей для своего решения столь много времени, что по его прошествии зашифрованная информация перестанет быть актуальной для лиц, использовавших $f(x)$ в качестве шифра.

На основе указанных принципов была предложена трудноразрешимая задача факторизации большого числа, которая была положена в основу первой, реально используемой, системы шифрования – RSA [5].

Далее рассмотрим основные направления применения криптосистем с открытым ключом.

Применение криптосистем с открытым ключом

Двигателем данного направления криптографии является, в первую очередь, практика. Стремительное развитие информационных систем ставит все новые и новые задачи перед разработчиками криптографических алгоритмов (протоколов). Классифицируем наиболее существенные побудительные мотивы развития криптографии с открытым ключом (не претендует быть исчерпывающей):

- Развитие телекоммуникационных систем и сетей различного назначения.
- Развитие глобальной сети Интернет.
- Развитие банковских систем, в том числе и пластиковых карт.

- Потребность мыслящего человека к познанию.

Выделяют следующие основные (глобальные, в самом широком смысле) направления применения криптографических преобразований с открытым ключом:

- зашифровывание и расшифровывание;
- выработка общего секрета или обмен ключами;
- наложение и проверка электронной цифровой подписи;
- аутентификация;
- «электронные деньги».

Каждому, из перечисленных направлений, присуща собственная процедура применения открытого или личного кличей.

В современных условиях, для решения задач защиты информации могут применяться криптографические алгоритмы, которые могут решать как одну, так и несколько задач (из выше перечисленных). Алгоритм RSA [2], например, с успехом позволяет решать задачи шифрования, обмена ключами и цифровой подписи. Однако все универсальное не лишено недостатков. Для успешного противостояния криптоаналитикам и повышения эффективности алгоритма, были предложены различные модификации RSA [2], решающие указанные задачи по отдельности.

Условия применения криптосистем с открытым ключом

В свое время Диффи и Хеллманом предложили, что существуют криптосистемы, которые содержат два более ключа (с открытым ключом), без предоставления доказательств. Однако или были указаны условия, которым следует удовлетворять таким криптосистемам:

1. Для стороны A (или B) процесс генерации ключевых пар: Pb_A и Pt_A (или Pb_B и Pt_B), не вызывает вычислительных трудностей.

2. Для отправителя A (или B), процесс зашифровывания не вызывает вычислительных трудностей, при наличии открытого ключа Pb_B и сообщения M: $C = E_{Pb_B}(M)$.

3. Для получателя B (или A), процесс расшифровывания не вызывает вычислительных трудностей, при наличии личного ключа Pt_B и полученного зашифрованного сообщения C': $M = D_{Pt_B}(C') = D_{Pt_B}\left[E_{Pb_B}(M)\right]$.

4. Для злоумышленника, процесс восстановления личного ключа Pt_B из открытого ключа Pb_B, является вычислительно неосуществимым.

5. Для злоумышленника, процесс восстановления оригинального сообщения M из имеющегося зашифрованного текста C открытого ключа Pb_B, является вычислительно неосуществимым.

6. Функции зашифровывания и расшифровывания, могут применяться в произвольном порядке: $M = E_{Pb_B}\left[D_{Pt_B}(M)\right]$. Данное условие не является необходимым.

Сегодня широко известны и получили применение в криптографических приложениях целый ряд теоретико-числовые задачи, которые позволяют строить односторонние функции с «секретом», их классификация приведена на рис. 2. Далее, в работе, постараемся детально остановиться на описании наиболее известных задачах в следующей последовательности:

1. Краткое описание задачи.

2. Оценка вычислительной и пространственной сложности криптоанализа.

3. Производительность/вычислительная сложность программной реализации.

Далее, в работе, под сложностью будет пониматься именно вычислительная сложность.

Отметим, что при оценке сложности решения той или иной задачи, по возможности, будет указываться сложность ее решения с использованием квантового компьютера. Данное замечание становиться достаточно существенным в свете последней

презентации канадской компании D-Wave Systems в 2007 году реально действующего квантового компьютера [72]. Разработчиками было отмечено, что криптоанализ является одним из основных направлений его применения.

При описании практической реализации, сделаем следующее предположение: оценка производится с учетом реализации на W-разрядном процессоре, при этом операции присвоения и управления потоком выполнения команд – не учитываются.

Постоянное развитие вычислительной техники и увеличение ее производительности, что подтверждается законом Мура, а также совершенствованию математических методов криптоанализа влечет периодическому пересмотру размеров ключевой информации.

На рис. 2 используются следующие аббревиатуры:

- ЦП – цифровая подпись.
- АСШ – асимметричное шифрование.
- ОК – обмен ключами.

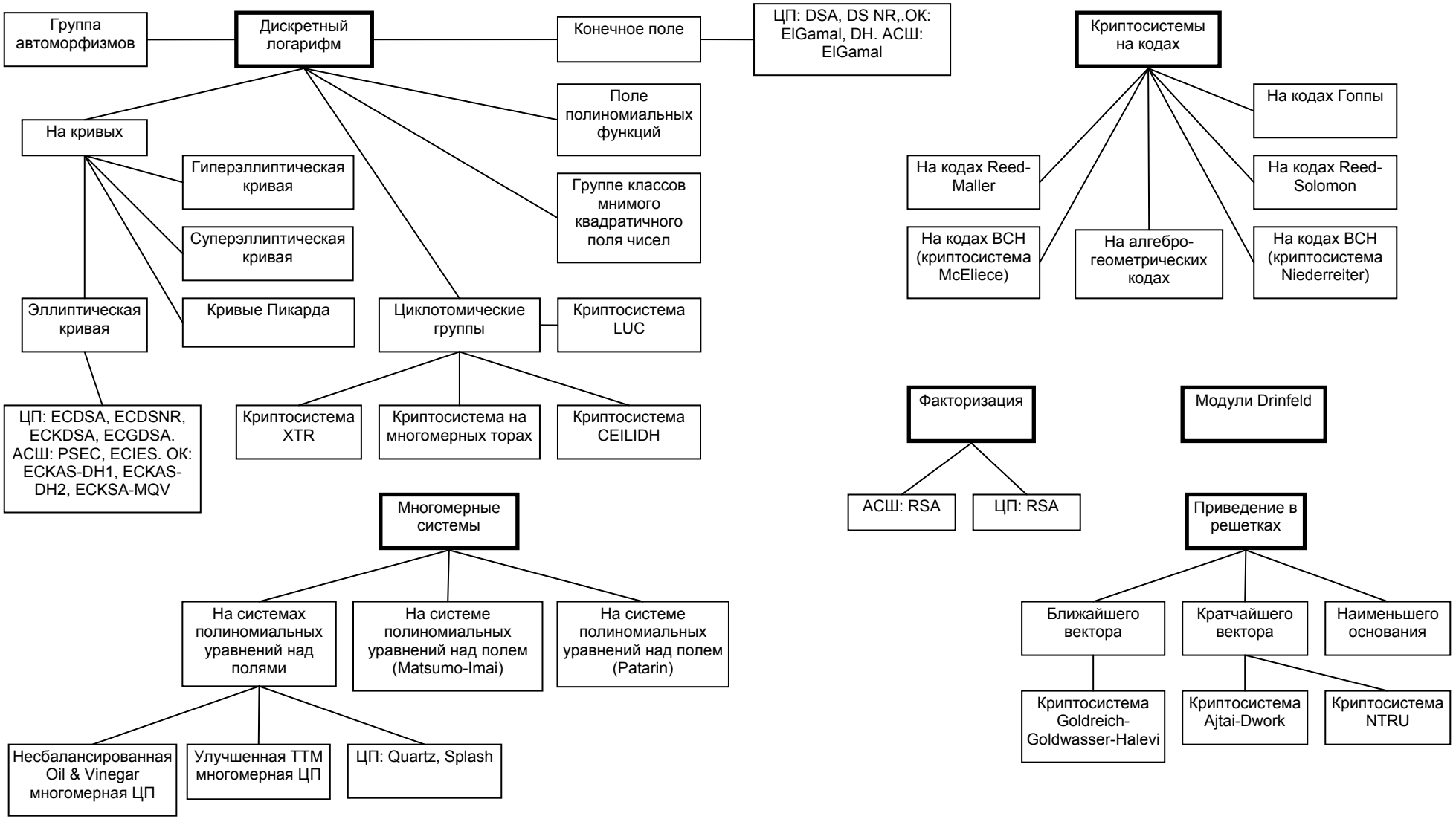

Рис. 2. Классификация теоретико-числовых задач, а также криптосистем на их основе

Последняя модификация: Vlad Kovtun

Дата последней модификации: 31.01.2011 21:24:00

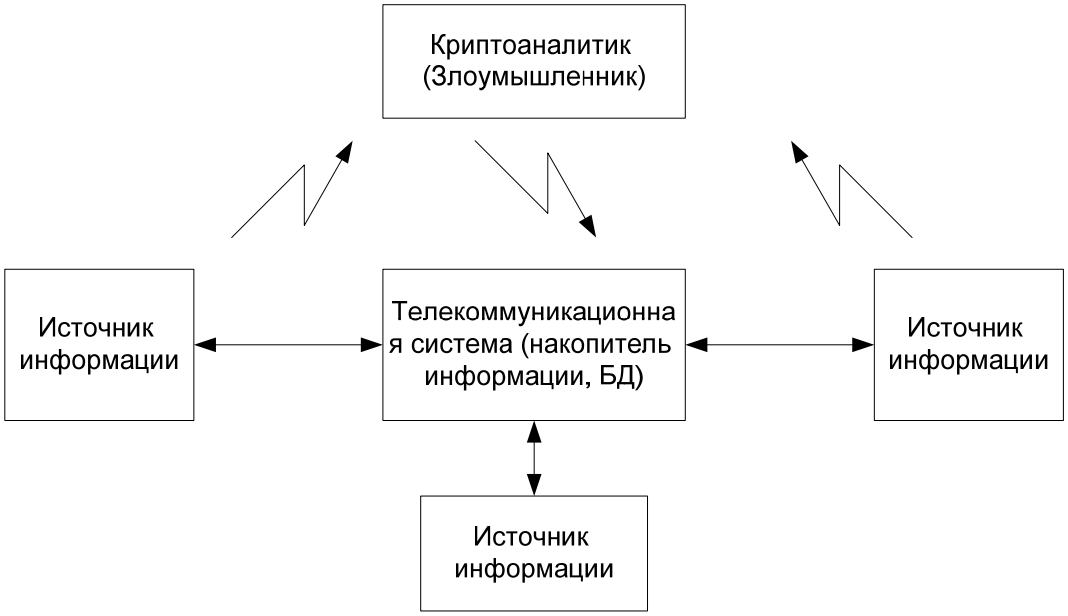

Рис. 3. Схема коммуникаций между множеством пользователей

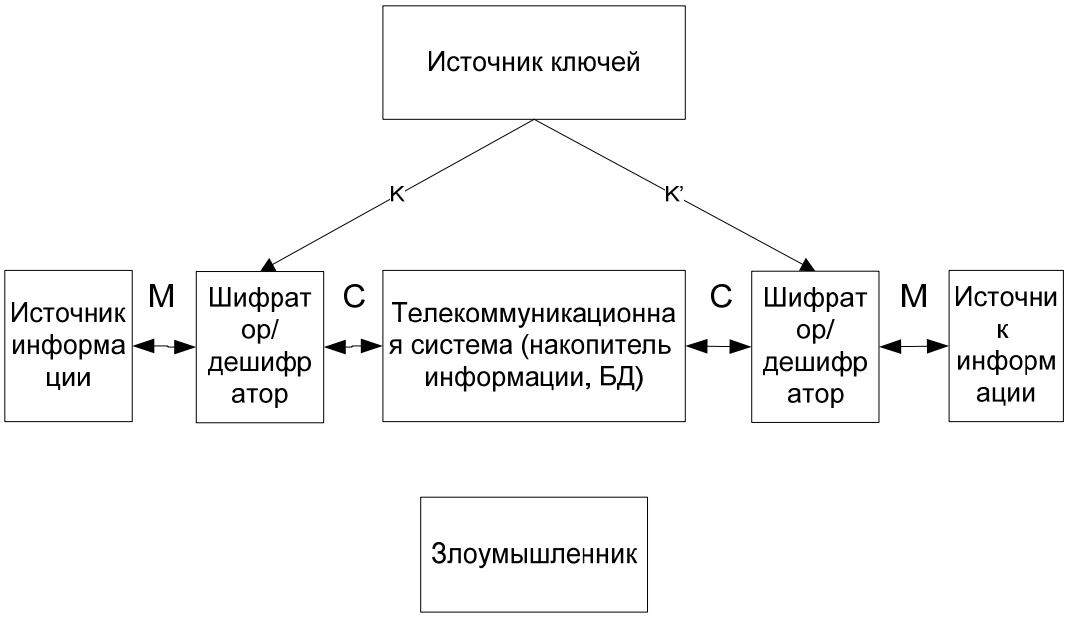

Рис. 4. Новая криптографическая схема защищенной передачи информации.

Расшифруем обозначения на рис. 4:

- Алфавит A, в котором записываются сообщения (открытые тексты), и алфавит B, в котором записываются криптотексты.
- Пространство ключей K (множество слов в некотором алфавите).
- Алгоритм **генерирования ключей**. Это полиномиальный (эффективный) вероятностный алгоритм, который выдает (позволяет сформировать) случайную пару $K,K' \in \mathrm{K}$. Компонента K называется **открытым ключом** и используется для шифрования, а компонента K' называется **личным ключом** и используется для дешифрования.
- Полиномиальный (эффективный) детерминированный алгоритм шифрования E, который получает на вход сообщения M и открытый ключ K, а выдает криптотекст C, что записывается как $C = E_K(M)$.

- Полиномиальный (эффективный) детерминированный алгоритм дешифрования D, который получает на вход криптотекст C и секретный ключ K', а выдает открытый текст M, что записываем как $M = D_{K'}(C)$

Перечисленные алгоритмы удовлетворяют таким условиям:

- Если пара (K,K') порождена алгоритмом генерирования ключей, то из $C = E_K(M)$ следует $M = D_{K'}(C)$ для любого открытого текста M.
- Нет (или, по крайней мере, неизвестно) хотя бы одного эффективного алгоритма, который бы по известным $C = E_K(M)$ и K находил бы M.

Последнее условие обеспечивает надежность криптосистемы даже тогда, когда из открытого ключа K не делается секрета. Из этого условия следует, в частности, что пара (K,K') могла бы быть порожденной алгоритмом генерирования ключей.

Заметим, что когда открытый ключ является доступным для злоумышленника, последний имеет практически те же возможности криптоанализа, которые для традиционных криптосистем назывались атакой с выбранным открытым текстом. Как и ранее, сильнейшим видом криптоанализа остается атака с выбранным криптотекстом.

Криптосистемы с открытым ключом еще называют и **асимметричными**. В этом контексте классические криптосистемы называют **симметричными**.

Криптосистема RSA

Эта система предложена в 1977 году, и является одной из наиболее популярных криптосистем с открытым ключом. Название системы образовано из первых букв имен ее создателей – Рональда Райвеста, Ади Шамира и Леонарда Адлемана.

Генерирование ключей. Выбирают два достаточно больших простых числа p и q. Для их произведения $n = pq$ функция Эйлера равна $\varphi(n) = (p-1)(q-1) = n - p - q + 1$ (в теории чисел используется понятие функции Эйлера $\varphi(m)$, под которой понимается число чисел меньших m и взаимно простых с m). Затем случайным образом выбирают число *e*, которое не превышает значения $\varphi(n)$ и взаимно простое с ним. Для этого числа *e* по алгоритму Евклида находят элемент d, обратный к e, т.е. такой, что $d < \varphi(n)$ и $ed \equiv 1(\mathrm{mod}\,\varphi(n))$.

Эта запись в теории чисел обозначает, что произведение ed при делении на число $\varphi(n)$ дает остаток равный 1 (читается ed сравнимо с единицей по модулю $\varphi(n)$).

В результате полагают:

- В качестве **открытого ключа** пару чисел e и n.
- В качестве **личного ключа** – число d.

Шифрование осуществляется блоками. Для этого сообщение записывают в цифровом виде и разбивают на блоки так, что каждый блок представляет число, которое не превышает n. Скажем, если блок M записан в двоичной форме длины m, то должно выполняться условие $2^m < n$.

Алгоритм шифрования E в системе RSA состоит в возведении двоичного числа M в степень e. Запишем его так: $E(M) = M^e(\mathrm{mod}\,n)$.

В результате получается шифроблок $C = E(M)$.

Дешифрование также осуществляется блоками. Алгоритм дешифрования D шифроблока C состоит в возведении числа C в степень d, т.е. $D(C) = C^d(\mathrm{mod}\,n)$.

Пример. Пусть $p = 53$ и $q = 67$. Тогда $n = 3551$ и $\varphi(n) = 3432$. Возьмем $e = 1021$можно проверить, что $\gcd(1021, 3432) = 1$. Одновременно вычисляем $d = 1021^{-1}(\mathrm{mod}\,3432) = 1237$. Ключи найдены.

Публикуем открытый ключ $e = 1021$ и $n = 3551$. Предположим, что один из деловых партнеров решил послать нам указание ПРОДАЙ. Сначала он преобразует свое

сообщение к цифровому виду, заменив каждую букву ее двух цифровым десятичным номером в алфавите 17 18 16 04 00 06 (здесь нумерация букв алфавита начинается с нуля). Видно, что для нашего модуля $n = 3551$ цифровое сообщение целесообразно разбивать на блоки по 4 цифры, т.е. 1718 1604 0006. При шифровании первый блок преобразуется в число $1718^{1021} \pmod{3531} = 139$. Таким же образом шифруются и остальные блоки, и в результате получается шифртекст 0139 ... .

Дешифрование этой шифрограммы выполняется возведением каждого блока в степень $d = 1237$ по модулю $n = 3551$. Легко проверить, что $139^{1237} \pmod{3551} = 1718$ и т.д.

Корректность. Следует убедиться, что $D(E(M)) = M$ для любого сообщения M. Сформулируем это в виде утверждения.

Утверждение 1. Пусть $n = pq$ является произведением двух разных простых чисел. Если $ed \equiv 1 \pmod{\varphi(n)}$, то для всех $x \in \mathbf{Z}_n$ справедливо сравнение $x^{ed} \equiv x \pmod{n}$.

Этот результат непосредственно следует из соответствующей теоремы Эйлера, которая утверждает, что для любого натурального числа $n > 1$ и числа x, взаимно простого с n, т.е. $(a, n) = 1$, справедливо сравнение $x^{\varphi(n)} \equiv 1 \pmod{n}$.

Действительно, в данном случае из $ed \equiv 1 \pmod{\varphi(n)}$ следует равенство $ed = k\varphi(n) + 1$, $k \in \mathbf{N}$ ($\mathbf{N}$-множество натуральных чисел), из которого в свою очередь следует справедливость $x^{ed} \equiv x \pmod{n}$.

Эффективность. Алгоритм генерирования ключей использует процедуру порождения простых чисел и расширенный алгоритм Евклида для вычисления $\gcd(e, \varphi(n))$ и $d = e^{-1} \bmod \varphi(n)$. В алгоритмах шифрования и дешифрования возведение в степень выполняется с помощью бинарного метода (либо других методов, например Лим-Лии).

Криптосистема Эль-Гамаля

Генерирование ключей. Выбирают большое простое число p, а также число g, $1 < g < p - 1$, которое имеет в мультипликативной группе $\mathbf{Z}_p^{\bullet}$ большой порядок. В идеальном случае g первообразным корнем по модулю p. Числа p и g являются не секретными и находятся в общем пользовании. Каждый абонент выбирает себе случайное число a в промежутке от 1 до $p - 1$, и вычисляет $h = g^a \bmod p$.

Открытый ключ: p, g, h.

Личный ключ: a.

Шифрование осуществляется блоками. Каждый блок M считается элементом из $\mathbf{Z}_p^{\bullet}$ (мультипликативная группа элементов, для которых в $\mathbf{Z}_n$ имеются обратные относительно умножения элементы (имеются мультипликативные инверсии)). Сообщение $M \in \mathbf{Z}_p^{\bullet}$ преобразуется в криптотекст $C \in \left(\mathbf{Z}_p^{\bullet}\right)^2$ следующим образом:

- Выбирается случайное число r такое, что $1 \le r \le p - 1$.
- Вычисляют $C = (c_1, c_2)$, где $c_1 = g^r \bmod p$, $c_2 = Mh^r \bmod p$.

Дешифрование. Имея секретный ключ a и криптотекст $C = (c_1, c_2)$, вычисляют $D(C) = c_2 \cdot \left(c_1^a\right)^{-1} \bmod p$.

Пример. Как и во всех предыдущих случаях, мы жертвуем реализмом для простоты вычислений. Пусть $p = 23$, $g = 5$, $a = 6$. Вычисляем $h = 5^6 \bmod 23 = 8$. Открытый и секретный ключ сформированы.

Предположим, что шифруется числовая информация и необходимо зашифровать сообщение $M = 7$. Пусть выбрано $r = 10$. Тогда $c_1 = 5^{10} \bmod 23 = 9$ и

$c_2 = \left(7 \cdot 8^{10}\right) \bmod 23 = 21$. Получаем криптотекст $C = (9,21)$. Легко проверить, что при дешифровании этого криптотекста действительно $D(9,\ 21) = 21 \cdot \left(9^6\right)^{-1} \bmod 23 = 7$.

Корректность. Проверка равенства $D(C) = M$ для криптотекста C, полученного из сообщения M с помощью алгоритма шифрования с произвольным r, проверяется непосредственно

$$D(C) = c_2 \cdot \left(c_1^a\right)^{-1} (\bmod p) = Mh^r \cdot \left(g^{ra}\right)^{-1} (\bmod p) = M\left(g^a\right)^r \left(g^{ra}\right)^{-1} (\bmod p) = M .$$

Идея криптосистемы достаточно прозрачна: сообщение M маскируется, приобретая вид c_2, и вместе с тем посылается подсказка c_1, которая позволяет воспроизвести M из c_2.

Эффективность. Возведение в степень в $\mathbf{Z}_p^*$ по модулю выполняется с помощью бинарного метода. Алгоритм выбора большого простого p относится к уже хорошо освоенным. Однако наша задача сложнее – необходимо выбрать также число g. В лучшем случае необходимо иметь в качестве g первообразный корень по модулю p. К сожалению, эта задача не имеет простого вычислительного решения. Поэтому следует ставить задачу генерирования пары p, g, для решения которой существуют эффективные процедуры.

Однонаправленные хэш-функции

Для обеспечения целостности информации используются хеш-функции. Наиболее полно требованиям, предъявляемым к криптографически стойким методам хеширования информации, отвечают однонаправленные хеш-функции. Данные хеш-функции определены в международном стандарте ISO/IEC 10118.

Хэш-функция **предназначена** для сжатия подписываемого документа M до нескольких десятков или сотен бит.

Хэш-функция $h(M)$ принимает в качестве аргумента сообщение M произвольной длины, а возвращает хэш-значение $h(M) = H$ фиксированной длины. Значение хэш-функции $h(M)$ сложным образом зависит от документа M и не позволяет восстановить сам документ M.

Хэш-функция должна удовлетворять ряду условий:

- должна быть чувствительна к всевозможным изменениям в тексте M, таким как вставки, выбросы, перестановки и т.п.;
- должна обладать свойством необратимости, т.е. задача подбора документа M', который обладал бы требуемым значением хэш-функции, должна быть вычислительно неразрешимой;
- вероятность того, что значение хэш-функций двух различных документов совпадут, должна быть ничтожно мала.

Большинство хэш-функций строится на основе однонаправленных функций.

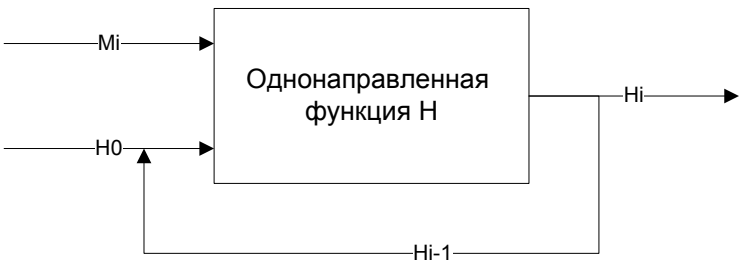

Рис. 5. Схема однонаправленной функции

M_i – блок исходного текста; H_{i-1} – хеш-значение предыдущего блока текста; H_i – хэш-значение последующего блока текста.

Хэш-значение, вычисляемое при вводе последнего блока текста становится хэш-значением всего сообщения M .

Существуют алгоритмы безопасного хэширования (SHA, разработан в 1992г., предназначен для использования совместно с алгоритмом цифровой подписи DSA. Отечественный стандарт хеш-функции ГОСТ Р34.11-94 базируется на блочном алгоритме шифрования ГОСТ 28147-89, используется совместно с российским стандартом цифровой подписи.

Стандарт ISO/IEC 10118 состоит из четырех частей. В **первой части** стандарта вводятся общие определения и понятия для других частей стандарта, а также модель итеративных хеш-функций.

Все хеш-функции, определенные в стандарте, как и большинство бесключевых хеш-функций, построены на основе итеративной модели. На рис.1. представлена общая модель итеративной хеш-функции.

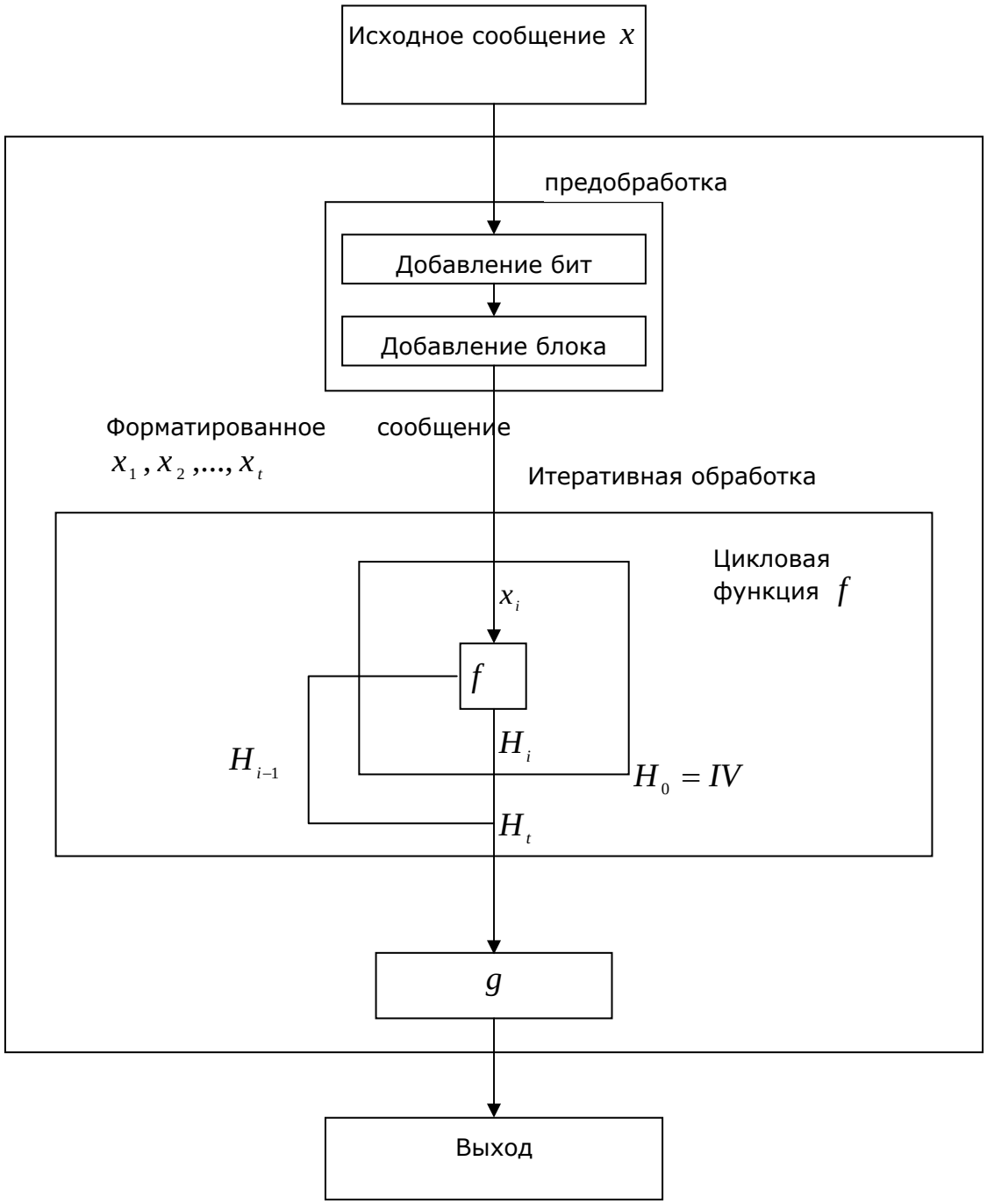

Рис. 6. Итеративная модель хеш-функции

Согласно данной модели, формирование хеш-значения осуществляется следующим образом. На вход хеш-функции h поступает исходное сообщение x – строка данных произвольной длины. Данная строка представляется в виде последовательности r-разрядных блоков x_i. Перед обработкой, по необходимости, последний блок дополняется до r битов. Кроме того, из соображений повышенной стойкости, исходная строка может быть дополнена новым r-разрядным блоком, который например, может содержать двоичное представление числа, характеризующего длину исходного сообщения x. Стандарт ISO/IEC 10118 не определяет методы дополнения, но предполагается, что могут быть использованы методы, определенные в ISO/IEC 9797.

Основой хеш-функции является так называемая цикловая функция $f(x, y)$ или функция сжатия, которая, в общем случае, берет на вход две строки x и y, длиной соответственно r и s разрядов и формирует на выходе s-разрядную строку. Каждый блок x_i служит входным аргументом для цикловой функции. Другим аргументом является s разрядное промежуточное значение (переменная сцепления), полученное на предыдущем шаге хеширования. H_i обозначает частный результат хеширования на i-ом шаге (итерации), IV - s-разрядный вектор инициализации. На выходе цикловой функции f формируется s-разрядное хеш-значение. При необходимости оно может быть усечено до заданной длины посредством дополнительной обработки $g(H_t)$. Чаще всего $g(H_t) = H_t$.

Во **второй части** стандарта определены хеш-функции, использующие n-разрядный алгоритм блочного шифрования.

Практической мотивацией построения таких хеш-функций является то, что использование в качестве основного компонента хеш-функции достаточно эффективной реализации блочного шифра, который применяется в системе, что позволяет обеспечить большую функциональность системы при меньших финансовых затратах.

В данной части стандарта определено два метода формирования хеш-значения. Первый метод позволяет вырабатывать хеш-значение однократной длины (64 бита), второй метод – хеш-значение двойной длины (128 бит). К хеш-функциям однократной длины относятся:

- Хеш-функция Матиаса-Майера-Озиса (Matyas-Mayer-Oseas).
- Хеш-функция Дэвиса-Майера (Davies-Meyer).
- Хеш-функция Миагуччи-Пренеля (Miyaguchi-Preneel).

Стойкость данных хеш-функций базируется на стойкости используемого в их основе блочного шифра, который должен обладать требуемыми свойствами случайности формируемых выходных значений. Кроме того, для данного алгоритма не должны реализовываться атаки, которые опираются на использование каких-либо особых свойств и особенностей структуры алгоритма.

Второй метод, определяемый стандартом, вырабатывает хеш-код двойной длины и опирается на алгоритм MDC-2. MDC-код (modification detection code), также известный как код обнаружения манипуляций над сообщениями, или код целостности сообщений, предназначен для формирования сжатого образа или хеш-значения сообщения, который удовлетворяет специальным средствам.

Данный метод выполняет две операции шифрования на обработку каждого входного блока. Стандарт ориентирован на использование в качестве основы алгоритма DES. Однако общая конструкция может быть использована и для построения хеш-функций на основе других блочных шифров.

На основе MDC-2 построена более сложная хеш-функция MDC-4. Хеш-функция MDC-4 в качестве цикловой функции использует хеш-функцию MDC-2. Одна итерация цикловой функции MDC-4 включает в себя последовательное выполнение двух цикловых функций MDC-2.

В **третьей части** стандарта определены специализированные хеш-функции (dedicated hash function). Стандарт описывает три типа таких хеш-функций, а именно американский стандарт SHA-1, и европейский стандарт RIPEMD-128, RIPEMD-160.

Специализированные хеш-функции (customized hash functions, или dedicated hash functions) специально разработаны только для целей хеширования и оптимизированы для выполнения данной задачи. Представленные в стандарте хеш-функции основаны на использовании принципов построения, заложенных в хеш-функции семейства MDx (MD2, MD4, MD5), которые специально разрабатывались для реализации на 32-разрядных ЭВМ. Алгоритм MD4 был предложен Р. Райвестом в 1990 году, а в 1991 тот же автор предложил модифицированную версию алгоритма MD5. В настоящее время хеш-функции MD4, MD5 являются наиболее распространенными в практических приложениях хеш-функциями. Однако некоторые их недостатки не позволили стандартизировать их.

Европейский консорциум RIPE, опираясь на свои исследования свойств этих алгоритмов, предложил усиленную версию MD4, которая получила название RIPEMD. Хеш-функция RIPEMD по сути состоит из двух параллельно работающих и модифицированных функций MD4 (т.е. функция имеет две линии). Суть модификации заключается в изменении аргументов операторов циклических сдвигов и порядка следования на вход циклов хеш-функции слов хешируемого сообщения. Параллельные линии, кроме того, отличаются использованием различных констант. Версии RIPEMD-128 и RIPEMD-160 вырабатывают хеш-код длиной 128 и 160 бит соответственно.

Другой альтернативой алгоритмам MDx является алгоритм SHA-1, разработанный совместно Агентством национальной безопасности США и NITS и принятый в качестве американского национального стандарта (FIPS 180-1).

По сравнению со 128-разрядными хеш-функциями, 160-разрядный хеш-код, вырабатываемый SHA-1, обеспечивает большую стойкость к силовым атакам. Хеш-функции SHA-1 и RIPEMD-160 по стойкости приблизительно равны и обе превосходят MD5. Расширение блока входного сообщения введено с целью обеспечения большего отличия между входными блоками. Вводимая избыточность способствует повышению стойкости хеш-функции.

В **четвертой части** стандарта определены хеш-функции, использующие модулярную арифметику. Стандарт вводит две хеш-функции MASH-1 и MASH-2, которые используют модульное возведение в степень для построения хеш-кода. Эти функции являются улучшеными вариантами функций, определенных в стандартах CCITT X.509:1988 и ISO/IEC 9594-8:1989, которые были признаны недостаточно безопасными. В следствии этого они были исключены из обновленных версий стандартов ITU-T X.509:1993 и ISO/IEC 9594-8:1995.

Основной идеей хеш-функций, основанных на модулярной арифметике, является использование в качестве цикловой функции итеративной функции, использующей модулярную арифметику. Причинами стандартизации и применения таких хеш-функций является, во-первых, возможность использования существующих программных и аппаратных средств модулярной арифметики, применяемых в несимметричных криптосистемах, и во-вторых, обеспечение требуемого уровня стойкости. Основным недостатком функций является низкая скорость формирования хеш-кода.

Хеш-функция MASH-1 (Modular Arithmetic Secure Hash) использует RSA подобные модули N, длина которых обеспечивает требуемую стойкость. Число N должно быть трудноразложимым, на чем и основывается стойкость алгоритма. Размер модуля определяет длину блоков обрабатываемого сообщения, а также размер хеш-кода. Алгоритм MASH-2 отличается от алгоритма MASH-1 только показателем степени в цикловой функции.

На рис. 5 представлены однонаправленные хеш-функции, разработанные в соответствии с используемыми подходами.

Анализируя представленные хеш-функции, можно отметить, что наибольшего быстродействия обработки информации позволяют достигать, естественно, специализированные хеш-функции как функции, изначально построенные по заказному принципу. Следствием такого подхода (то есть специальной разработки хеш-функций «с самого начала» для хеширования информации) является оптимизция алгоритмов хеширования по эффективности, отсутствие ограничений по многократному использованию существующего системного компонента, таких как блочные шифры или модулярная арифметика. Данные функции изначально подразумевают программно-ориентированную реализацию и оптимизацию итеративных процедур хеширования, следствием чего, помимо криптографической стойкости, является высокая скорость преобразования информации. В силу данных причин хеш-функции, построенные на основе MD – технологии, нашли широкое применение в телекоммуникационных

системах и могут быть эффективно реализованы не только на специализированных микропроцессорах, как, например, хеш-функции, основанные на модулярной арифметике, но и микропроцессорах общего назначения, широко используемых в телекоммуникационных системах. В табл.1, 2 представлены основные характеристики хеш-функций, построенных на основе MD – технологии.

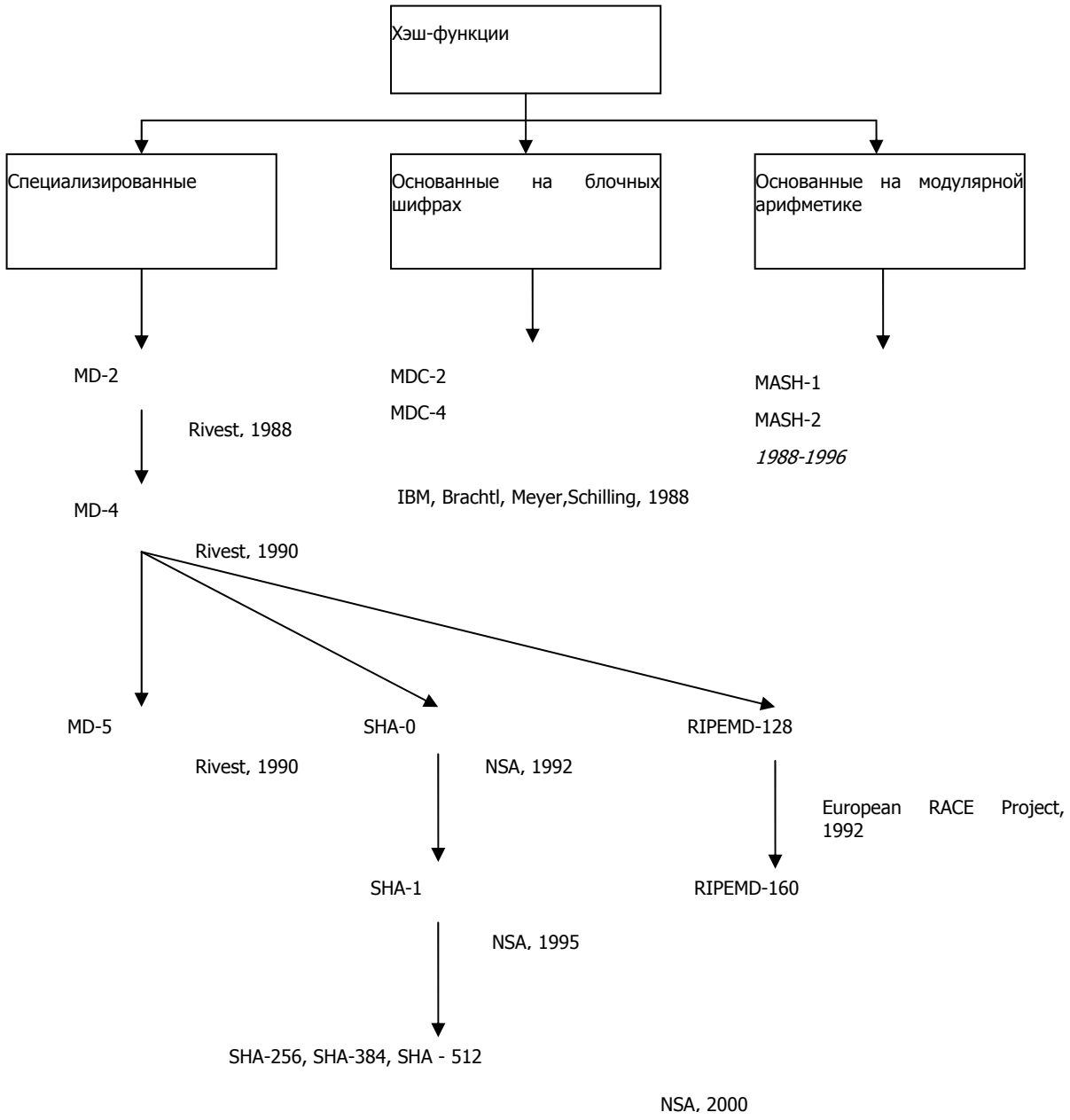

Рис. 7. Хеш-функции, хеш-функции, разработанные в соответствии со стандартом ISO/IEC 10118

Таблица 1. Характеристики хеш-функций, часть 1

Алгоритм	Общее кол-во шагов, *r•s*	Кол-во циклов, *r*	Кол-во шагов в цикле, *s*	Константы, отличные для каждого
MD-5	64	4	16	шаг
RIPEMD-128	64	4	16	цикл
RIPEMD-160	80	5	16	цикл
SHA-1	80	4	20	цикл
SHA-256	64	1	64	шаг

SHA-384	80	1	80	шаг
SHA-512	80	1	80	шаг

Хеш-функции, построенные с использованием блочных шифров и модулярной арифметики, обладают сравнительно меньшим быстродействием, причем наименьшим быстродействием обладают хеш-функции, построенные с использованием модулярной арифметики. Отметим, что последний класс функций большого распространения не получил в силу низкой вычислительной эффективности и необходимости использования специализированного программного и аппаратного обеспечения, что обуславливает их высокую себестоимость при реализации в системах защиты информации.

Таблица 2. Характеристики хеш-функций, часть 2

Алгоритм	Размер выхода, *бит*	Размер слова, *бит*	Кол-во слов на выходе	Кол-во слов для промежуточного значения
MD-5	128	32	4	4
RIPEMD-128	128	32	4	8
RIPEMD-160	160	32	5	10
SHA-1	160	32	5	5
SHA-256	256	32	8	8
SHA-384	384	64	8	8
SHA-512	512	64	8	8

Следует отметить, что специализированные хеш-функции более других хеш-функций подвержены методам криптоанализа. Это обусловлено тем, что стойкость данных функций определяется, в первую очередь, стойкость к статистическим методам анализа: цикловая функция в процессе обработки информации в каждой итерации использует специальным образом подобранные константы и определенные функции с фиксированными линейными и нелинейными операциями. Другими словами, стойкость таких функций основывается не столько на доказуемой стойкости какого-либо хорошо зарекомендовавшего подхода, а тщательной и скрупулезной разработке каждой используемой функции, длительном подборе используемых констант, число которых достигает нескольких десятков. Оценка стойкости осуществляется с использованием различных статистических тестов. Недостатком данного подхода является, во-первых, то, что ошибка или какое-либо упущение при выборе используемых цикловых констант и их последующем микшировании разработанными функциями может критическим образом повлиять на стойкость функции, и, во-вторых, то, что используемые статистические методы, обеспечивающие стойкость на сегодняшний день, могут оказаться недостаточными на следующий день.

В то же время хеш-функции, построенные с использованием блочных шифров, являются потенциально более стойкими, поскольку их стойкость базируется на стойкости хорошо проверенных и зарекомендовавших себя блочных шифров, а в более общем плане – на стойкости разработанных ранее общих концептуальных принципах (подходах). Кроме того, данные функции (как «*образы*» блочных шифров) основаны на использовании таких основополагающих криптографических принципов, как замешивание и рассеивание: такие функциональные составляющие, как блоки подстановки, реализуют замешивание за счет введения нелинейности, а блоки линейного рассеивания реализуют рассеивание за счет использования специальным образом подобранных матриц. Анализ замешивания и рассеивания затруднений не представляет в силу хорошо известной и используемой математической базы; в специализированных же хеш-функциях степень замешивания и рассеивания представляется затруднительной и оценивается только с помощью статистических тестов и носит условный характер.

На рис.8 представлена стойкость специализированных функций, построенных на основе MD – технологий.

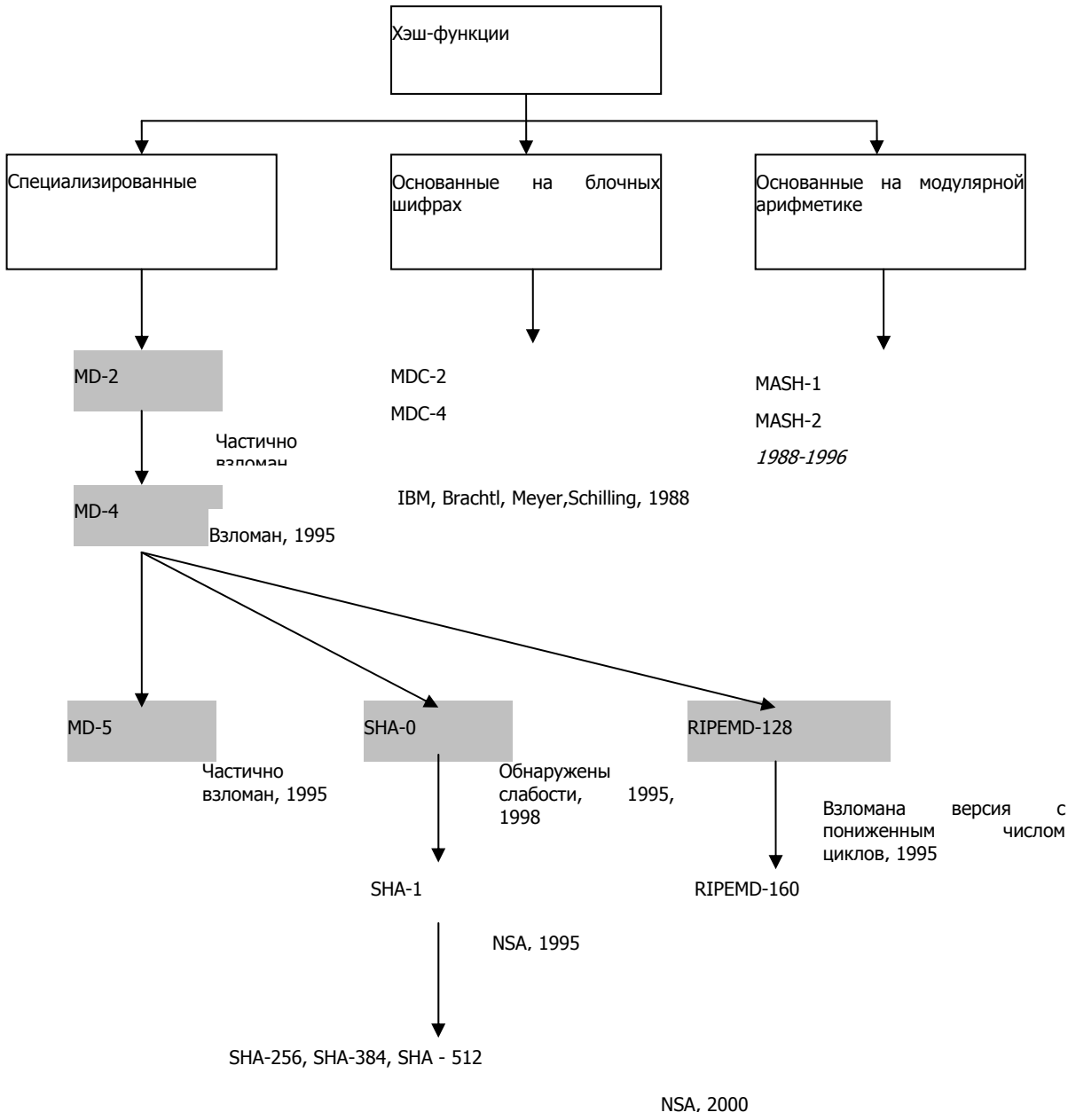

Рис. 8. Стойкость специализированных хеш-функций, построенных на основе MD – технологий

Как видно из приведенных данных, специализированные хеш-функции в достаточной степени подвержены методам криптоанализа. Повышение стойкости таких функций базируется не на повышении стойкости функциональных составляющих алгоритма, а на увеличении количества шагов в циклах и размеров обрабатываемых блоков данных, что обуславливает понижение быстродействия обработки информации.

Таким образом, с точки зрения обеспечения стойкости к методам анализа наиболее целесообразным может являться использование хеш-функций, основанных на использовании хорошо зарекомендовавших себя блочных шифров. Данные шифры являются хорошо изученными, стандартизированными, существует обширный инструментарий, предназначенный для анализа стойкости данных шифров. Кроме того, отпадает необходимость в дополнительных финансовых затратах, связанных с разработкой специализированных хеш-функций и их внедрением в системы защиты информации как избыточной, с точки зрения стойкости, составляющей – безопасность информации в данных системах обеспечивается именно за счет применения блочных шифров.

Литература

1. Криптографическая защита информации в информационных системах. Курс лекций. И.Д. Горбенко. ХНУРЭ. 2002.

2. Брюс Шнайер. Прикладная криптография. 2-ое издание. Протоколы, алгоритмы и исходные тексты на языке С. Доступно: http://nrjetix.com/r-and-d/lectures

3. Diffie W., Hellman M. New directions in cryptography // IEEE Trans. Inform. Theory, Vol 22, 6 (1976). P. 644–654.

4. Diffie W., Hellman M. E. "Multi-user Cryptographic Techniques", Proceedings of AFIPS National Computer Conference. -1976. -pp.109-112.

5. Rivest R., Shamir A., Adleman L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems // Communications of the ACM. -Vol. 21 (2). -1978. -pp.120–126.

6. Merkle R. C. "Secure Communication Over Insecure Channels", Communications of the ACM, v.21, n.4, 1978, pp.294-299.

7. Simmons G. I. "Cryptology", Encyclopedia Britannica, 16th edition, 1986, pp.913-924B.

8. Варновский Н.П. Криптография и теория сложности // Математическое просвещение. –Сер. 3. –Вып. №2, –1998. –С. 71–86.

9. Журнал Компьютерра, #1, 1998, с. 6.