11. АСИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ

11.1 Концепция криптосистемы с открытым ключом

Эффективными системами криптографической защиты данных являются асимметричные криптосистемы, называемые также криптосистемами с открытым ключом. В таких системах для зашифрования данных используется один ключ, а для расшифрования - другой ключ (отсюда и название-асимметричные). Первый ключ является открытым и может быть опубликован для использования всеми пользователями системы, которые зашифровывают данные. Расшифрование данных с помощью открытого ключа невозможно.

Для расшифрования данных получатель зашифрованной информации использует второй ключ, который является *секретным.* Разумеется, ключ расшифрования не может быть определен из ключа зашифрования.

Обобщенная схема асимметричной криптосистемы с открытым ключом показана на рис. 11.1.

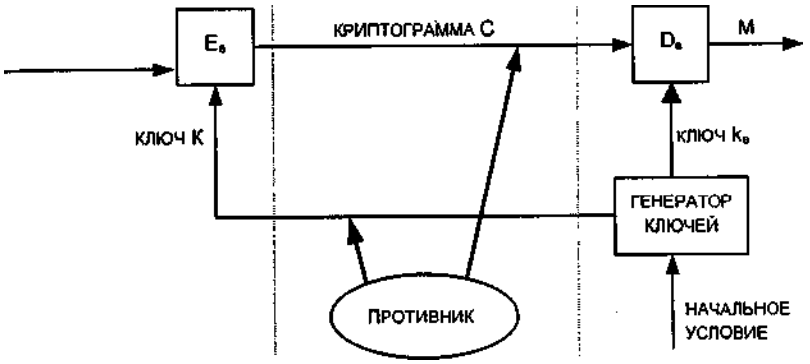

Рис. 11.1 - Обобщенная схема асимметричной криптосистемы

В этой асимметричной криптосистеме применяют два различных ключа:

- $K_в$ - *открытый ключ отправителя А;*
- $к_в$ - *секретный ключ получателя В.*

Генератор ключей целесообразно располагать на стороне получателя *В* (чтобы не пересылать секретный ключ $к_в$ по незащищенному каналу). Значения ключей $K_в$ и $к_в$ зависят от начального состояния генератора ключей.

Раскрытие секретного ключа $к_в$ по известному открытому ключу $K_в$ должно быть вычислительно неразрешимой задачей.

Характерные особенности асимметричных криптосистем:

1. Открытый ключ $K_в$ и криптограмма С могут быть отправлены по незащищенным каналам, т.е. противнику известны $K_в$ и С.

2. Алгоритмы шифрования и расшифрования являются открытыми: $E_в : M \rightarrow C$, $E_в^{-1} : C \rightarrow M$. Защита информации в асимметричной криптосистеме основана на секретности ключа $к_в$.

У. Диффи и М. Хеллман сформулировали требования, выполнение которых обеспечивает безопасность асимметричной криптосистемы:

1. Вычисление пары ключей $(К_в, k_в)$ получателем B на основе начального условия должно быть простым.
2. Отправитель A, зная открытый ключ $К_в$ и сообщение M, может легко вычислить криптограмму
$$C = EК_в(M) .$$
3. Получатель B, используя секретный ключ $k_в$ и криптограмму C, может легко восстановить исходное сообщение
$$= Ek_в \ (C) = Ekв \ (C) = Ekв \ [E_{Кв} (M)].$$
4. Противник, зная открытый ключ $К_в$, при попытке вычислить секретный ключ $k_в$ наталкивается на непреодолимую вычислительную проблему. Противник, зная пару $(К_в, C)$, при попытке вычислить исходное сообщение M наталкивается на непреодолимую вычислительную проблему.

11.2 Однонаправленные функции

Концепция асимметричных криптографических систем с открытым ключом основана на применении однонаправленных функций.

Неформально однонаправленную функцию можно определить следующим образом. Пусть $\boldsymbol{X}$ и $\boldsymbol{Y}$ - некоторые произвольные множества. Функция

$$\boldsymbol{f : X \rightarrow Y}$$

является однонаправленной, если для всех $\boldsymbol{xGX}$ можно легко вычислить функцию

И в то же время для большинства yeY достаточно сложно получить значение $\boldsymbol{xeX}$, такое, что $\boldsymbol{f^{-1}(x) = y}$ (при этом полагают, что существует по крайней мере одно такое значение $\boldsymbol{x}$).

Основным критерием отнесения функции $\boldsymbol{f}$ *к* классу однонаправленных функций является отсутствие эффективных алгоритмов обратного преобразования $\boldsymbol{Y \rightarrow X}$.

В качестве первого примера однонаправленной функции рассмотрим целочисленное умножение. Прямая задача - вычисление произведения двух очень больших целых чисел ***P*** и ***Q***, т.е. нахождение значения

$$N = P \cdot Q,$$

является относительно несложной задачей для ЭВМ.

Обратная задача-разложение на множители большого целого числа, т.е. нахождение делителей ***P*** и ***Q*** большого целого числа $N = P \cdot Q$, является практически неразрешимой задачей при достаточно больших значениях **N.** По современным оценкам теории чисел при целом $N \approx 2$ и $P \approx Q$ для разложения числа ***N*** потребуется около 10 операций, т.е. задача практически неразрешима на современных ЭВМ.

Следующий характерный пример однонаправленной функции - это модульная экспонента с фиксированными основанием и модулем. Пусть ***A*** и *N*-целые числа, такие, что $1 \le A < N$. Определим множество Z_N :

$$Z_N = \{0,1, 2, \dots , N-1\}.$$

Тогда модульная экспонента с основанием A по модулю N представляет собой функцию

$$f_{AN} : Z_N \rightarrow Z_N ,$$

$$f_{AN}(x) = A^x \bmod N,$$

где ***X***- целое число, $1 < x < N-1$; операция $i \bmod j$ - остаток от целочисленного деления ***i*** на ***j*** .

Существуют эффективные алгоритмы, позволяющие достаточно быстро вычислить значения функции $f_{A,N}(x)$.

Если $y = A^x$, то естественно записать $x = \log_A(y)$.

Поэтому задачу обращения функции $f_{AN}(x)$ называют задачей нахождения дискретного логарифма или задачей дискретного логарифмирования.

Задача дискретного логарифмирования формулируется следующим образом. Для известных целых ***A, N, y*** найти целое число ***x***, такое, что

$$A^x \bmod N = y.$$

Алгоритм вычисления дискретного логарифма за приемлемое время пока не найден. Поэтому модульная экспонента считается однонаправленной функцией.

современным оценкам теории чисел при целых числах $A \approx 2$ и $N \approx 2$ решение задачи дискретного логарифмирования (нахождение показателя степени x для известного y) потребует около 10 операций, т.е.

эта задача имеет в 10 раз большую вычислительную сложность, чем задача разложения на множители. При увеличении длины чисел разница в оценках сложности задач возрастает.

Следует отметить, что пока не удалось доказать, что не существует эффективного алгоритма вычисления дискретного логарифма за приемлемое время. Исходя из этого, модульная экспонента отнесена к однонаправленным функциям условно, что, однако, не мешает с успехом применять ее на практике.

Вторым важным классом функций, используемых при построении криптосистем с открытым ключом, являются так называемые однонаправленные функции с «потайным ходом» (с лазейкой). Дадим неформальное определение такой функции. Функция

$$f : X \rightarrow Y$$

относится к классу однонаправленных функций с **«потайным ходом»** в том случае, если она является однонаправленной и, кроме того, возможно эффективное вычисление обратной функции, если известен «потайной ход» (секретное число, строка или другая информация, ассоциирующаяся с данной функцией).

В качестве примера однонаправленной функции с «потайным ходом» можно указать используемую в криптосистеме RSA модульную экспоненту с фиксированными модулем и показателем степени. Переменное основание модульной экспоненты используется для указания числового значения сообщения *M* либо криптограммы *C*.

11.3 Криптосистема шифрования данных RSA

Алгоритм RSA предложили в 1978 г. три автора: Р. Райвест (Rivest), А. Шамир (Shamir) и А. Адлеман (Adleman). Алгоритм получил свое название по первым буквам фамилий его авторов. Алгоритм RSA стал первым полноценным алгоритмом с открытым ключом, который может работать как в режиме шифрования данных, так и в режиме электронной цифровой подписи.

Надежность алгоритма основывается на трудности факторизации больших чисел и трудности вычисления дискретных логарифмов.

Введем следующие понятия:

1. Простое число - делится только на 1 и на само себя;
2. Взаимно простым- не имеют ни одного общего делителя, кроме 1;
3. Результат операции $i \bmod j$ - остаток от целочисленного деления i на

В криптосистеме RSA открытый ключ $K_в$, секретный ключ $к_в$, сообщение *M* и криптограмма C принадлежат множеству целых чисел

$$Z_N = \{0, 1, 2, \ldots, N-1\},$$ где N - модуль: $N = P \cdot Q$.

Здесь ***P*** и ***Q*** - случайные большие простые числа. Для обеспечения максимальной безопасности выбирают ***P*** и ***Q*** равной длины и хранят в секрете.

Множество Z_N с операциями сложения и умножения по модулю ***N*** образует арифметику по модулю ***N***.

Открытый ключ $K_в$ выбирают случайным образом так, чтобы выполнялись условия:

$$1 < K_B < (p(N), НОД(K_B, ср(N)) = 1, <p(N) = (P\text{-}1)(Q \sim\sim 1),$$

где: ***φ(N)*** - функция Эйлера; **НОД** - наибольший общий делитель.

Функция Эйлера ***φ(N)*** указывает количество положительных целых чисел в интервале от 1 до **N**, которые взаимно просты с ***N***.

Второе из указанных выше условий означает, что открытый ключ $K_в$ и функция Эйлера ***φ(N)*** должны быть взаимно простыми.

Далее, вычисляют секретный ключ $k_в$, такой, что:

$$(k_B \cdot K_B) \bmod \varphi(N) = 1$$

или

$$K_в^{1} \bmod ((P-1)(Q-1)).$$

Это можно осуществить, так как получатель *B* знает пару простых чисел ***(P, Q)*** и может легко найти ***φ(N)***. Заметим, что $k_в$ и ***N*** должны быть взаимно простыми.

Открытый ключ $K_в$ используют для шифрования данных, а секретный ключ $k_в$ - для расшифрования.

Преобразование шифрования определяет криптограмму **C** через пару (открытый ключ $K_в$, сообщение ***M)*** в соответствии со следующей формулой:

$$C = E_{KB}(M) = (M^{KB}) \bmod N$$

В качестве алгоритма быстрого вычисления значения ***C*** используют ряд последовательных возведений в квадрат целого ***M*** и умножений на ***M*** с приведением по модулю ***N***.

Обращение функции $C = M^{KB} \bmod N$, т.е. определение значения ***M*** по известным значениям ***C***, $K_в$ и ***N***, практически не осуществимо при $N \approx 2$

Однако обратную задачу, т.е. задачу расшифрования криптограммы ***C***, можно решить, используя пару (секретный ключ $k_в$, криптограмма ***C)*** по следующей формуле:

$$= E_k\, yC) = 1C \quad \text{jmod(N)}$$

Таким образом, получатель В, который создает криптосистему, защищает два параметра:

- секретный ключ $k_в$
- пару чисел ***(P, Q)***, произведение которых дает значение модуля *N*.

С другой стороны, получатель *B* открывает значение модуля *N* и открытый ключ $K_в$.

Противнику известны лишь значения $K_в$ и ***N***. Если бы он смог разложить число ***N*** на множители ***P*** и ***Q***, то он узнал бы «потайной ход» - тройку чисел $\{P, Q, K_в\}$, вычислил значение функции Эйлера

$$\varphi(N) = (P-1)(Q-1)$$

и определил значение секретного ключа $k_в$. Однако, как уже отмечалось, разложение очень большого ***N*** на множители вычислительно не осуществимо (при условии, что длины выбранных ***P*** и ***Q*** составляют не менее 100 десятичных знаков).

11.3.1 Процедуры шифрования и расшифрования в криптосистеме RSA

Предположим, что пользователь *A* хочет передать пользователю *B* сообщение в зашифрованном виде, используя криптосистему RSA. В таком случае пользователь *A* выступает в роли отправителя сообщения, а пользователь *B* - в роли получателя. Как отмечалось выше, криптосистему RSA должен сформировать получатель сообщения, т.е. пользователь *B*. Рассмотрим последовательность действий пользователя *B* и пользователя *A*

1. Пользователь *B* выбирает два произвольных больших простых числа ***P*** *и* ***Q***.
2. Пользователь *B* вычисляет значение модуля $N = P \cdot Q$.
3. Пользователь *B* вычисляет функцию Эйлера: $\varphi(N) = (P-1)(Q-1)$ и выбирает случайным образом значение большого случайного числа, которое назовем $k_в$. Это число должно быть взаимно простым с функцией Эйлера (т.е. результатом умножения $\varphi(N) = (P-1)(Q-1)$)
4. Определяется такое число K_B, для которого является истинным следующее соотношения

$$(K_B \cdot k_в) \bmod (\varphi(N)) = 1$$

и

$$1 < K_B \le \varphi(N).$$

5. Пара чисел $(N, K_в)$ является открытым ключом и может быть передана по незащищенному каналу. А пара чисел $(N, k_в)$ - закрытым ключом, он держится в секрете и используется для дешифрации.

Если пользователь *A* хочет передать пользователю *B* сообщение ***M***, он выполняет следующие шаги.

6. Пользователь *A* разбивает исходный открытый текст ***M*** на блоки (только до ***N-1)***, каждый из которых может быть представлен в виде числа

$$M_i = 0, 1, 2, \dots, N-1.$$

7. Пользователь *A* шифрует текст, представленный в виде последовательности чисел ***Mi*** по формуле

$$C_i=(M_i^{KB}) \bmod (N)$$

и отправляет криптограмму

$$C0C1, \dots C_{N-1}$$

8. Чтобы расшифровать эти данные, используя секретный ключ $(N, k_в)$, необходимо выполнить следующие вычисления:

$$M_i=(C_i^{kB}) mod (N)$$

В результате будет получена последовательность чисел ***Mi***, которые представляют собой исходное сообщение ***M***. Чтобы алгоритм RSA имел практическую ценность, необходимо иметь возможность без существенных затрат генерировать большие простые числа, уметь оперативно вычислять значения ключей $K_в$ *и* $k_в$.

11.3.2 Пример использования алгоритма RSA

Зашифруем и расшифруем сообщение «CAB» по алгоритму RSA. Для простоты будут использованы маленькие числа. На практике применяются очень большие числа (см. следующий раздел).

1. Выберем $P = 3$ *и* $Q = 11$.

2. Определим $N = P \cdot Q = 3 \cdot 11 = 33$.

3. Найдем $\varphi(N) = (P-1)(Q-1) = (3-1)(11-1) = 20$.

Выбираем случайным образом значение числа $k_в$. Это число должно быть взаимно простым с функцией Эйлера (т.е. у $\varphi(N) = 20$ и $k_в$ не должно быть общих делителей кроме 1). Пусть $k_в = 3$.

4. Выберем число K_B по следующей формуле:

$(k_B \cdot K_B) \bmod 20 = 1$, т. е. произведение $k_B \cdot K_B$ при целочисленном делении на $\varphi(N) = 20$ должно в остатке давать 1.

Пусть $K_B = 7$ т.к.: 7·3=21 и (21 mod 20) = 1

$$\begin{array}{r|l} 21 & 20 \\ \hline 20 & 1 \\ \hline 1 & \end{array}$$

5. Представим шифруемое сообщение как последовательность чисел в диапазоне от 0 до 32 (кончается на ***N*-1**). Буква ***A*** =1, ***B***=2, ***C***=3.

6. Зашифруем сообщение, используя открытый ключ (K_B=7, N=33):

$C_1 = 3^7 \bmod 33 = 2187 \bmod 33 = 9$;

$$\begin{array}{r|l} 2187 & 33 \\ \hline 2178 & 66 \\ \hline 9 & \end{array}$$

$C_2 = 1^7 \bmod 33 = 1 \bmod 33 = 1$;

$$\begin{array}{r|l} 1 & 33 \\ \hline 0 & 0 \\ \hline 1 & \end{array}$$

$C_3 = 2^7 \bmod 33 = 128 \bmod 33 = 29$;

$$\begin{array}{r|l} 128 & 33 \\ \hline 99 & 3 \\ \hline 29 & \end{array}$$

Т.е. криптограмма представляет собой C = 09 01 29

7. Расшифруем эти данные, используя закрытый ключ (N=33, $k_в$=3).

$M_1=9^3 \bmod 33 = 729 \bmod 33 = 3$ (*C*);

$$\begin{array}{r|l} 729 & 33 \\ \hline 726 & 22 \\ \hline 3 & \end{array}$$

$M_2=1^3 \bmod 33 = 1 \bmod 33 = 1$ (*A*);

$$\begin{array}{r|l} 1 & 33 \\ \hline 0 & 0 \\ \hline 1 & \end{array}$$

$M_3=29^3 \bmod 33 = 24389 \bmod 33 = 2$ (*B*);

$$\begin{array}{r|l} 24389 & 33 \\ \hline 24387 & 739 \\ \hline 2 & \end{array}$$

Данные расшифрованы.

11.3.3 Безопасность и быстродействие криптосистемы RSA

Безопасность алгоритма RSA базируется на трудности решения задачи факторизации больших чисел, являющихся произведениями двух больших

простых чисел. Действительно, криптостойкость алгоритма RSA определяется тем, что после формирования секретного ключа $k_в$ и открытого ключа $K_в$ «стираются» значения простых чисел ***P и Q***, и тогда исключительно трудно определить секретный ключ $k_в$ по открытому ключу $K_в$, поскольку для этого необходимо решить задачу нахождения делителей ***P*** и ***Q*** модуля ***N***.

Разложение величины *N* на простые множители ***P*** и ***Q*** позволяет вычислить функцию ***φ(N) = (P-T) (Q-T)*** и затем определить секретное значение k_B используя уравнение

$$(K_B \cdot k_в) \bmod (\varphi(N)) = 1.$$

Другим возможным способом криптоанализа алгоритма RSA является непосредственное вычисление или подбор значения функции ***φ(N) = (P -T) (Q-T)***. Если установлено значение ***φ(N),*** то сомножители ***P*** и ***Q*** вычисляются достаточно просто. В самом деле, пусть

$$x = P + Q = N+1-<p(N),$$

$$y=(P-Q)^a = (P+G)^2-4*N.$$

Зная ***φ(N),*** можно определить ***x*** и затем ***y;*** зная ***x и y,*** можно определить числа ***P*** и ***Q*** из следующих соотношений:

$$P = 1/2(x+7y),\ 0 = 1/2(x-7y).$$

Однако эта атака не проще задачи факторизации модуля ***N***.

Задача факторизации является трудно разрешимой задачей для больших значений модуля ***N***.

Сначала авторы алгоритма RSA предлагали для вычисления модуля ***N*** выбирать простые числа ***P*** и ***Q*** случайным образом, по 50 десятичных разрядов каждое. Считалось, что такие большие числа ***N*** очень трудно разложить на простые множители. Один из авторов алгоритма RSA, P. Райвест, полагал, что разложение на простые множители числа из почти 130 десятичных цифр, приведенного в их публикации, потребует более 40 квадриллионов лет машинного времени. Однако этот прогноз не оправдался из-за сравнительно быстрого прогресса компьютеров и их вычислительной мощности, а также улучшения алгоритмов факторизации.

Один из наиболее быстрых алгоритмов, известных в настоящее время, алгоритм NFS (Number Field Sieve) может выполнить факторизацию большого числа ***N*** (с числом десятичных разрядов больше 120) за число

$$e^{2(1пл)1^{/3}(1п(1пл))2'з}$$

шагов, оцениваемых величиной

В 1994 г. было факторизовано число со 129 десятичными цифрами. Это удалось осуществить математикам А. Ленстра и М. Манасси посредством организации распределенных вычислений на 1600 компьютерах, объединенных сетью, в течение восьми месяцев. По мнению А. Ленстра и М. Манасси, их работа компрометирует криптосистемы RSA и создает большую угрозу их дальнейшим применениям. Теперь разработчикам криптоалгоритмов с открытым ключом на базе RSA приходится избегать

применения чисел длиной менее 200 десятичных разрядов. Самые последние публикации предлагают применять для этого числа длиной не менее 250-300 десятичных разрядов.

Оценка безопасных длин ключей асимметричных криптосистем на ближайшие 20 лет исходя из прогноза развития компьютеров и их вычислительной мощности, а также возможного совершенствования алгоритмов факторизации приведена в таблице 11.1 даны (для трех групп пользователей - индивидуальных пользователей, корпораций и государственных организаций), в соответствии с различием требований к их информационной безопасности. Конечно, данные оценки следует рассматривать как сугубо приблизительные, как возможную тенденцию изменений безопасных длин ключей асимметричных криптосистем со временем.

Таблица 11.1 - Оценки длин ключей для асимметричных криптосистем, бит

Год	Отдельные пользователи	Корпорации	Государственные организации
1995	768	1280	1536
2000	1024	1280	1536
2005	1280	1536	2048
2010	1280	1536	2048
2015	1536	2048	2048

11.4 Аутентификация данных и электронная цифровая подпись

При обмене электронными документами по сети связи существенно снижаются затраты на обработку и хранение документов, убыстряется их поиск. Но при этом возникает проблема аутентификации автора документа и самого документа, т.е. установления подлинности автора и отсутствия изменений в полученном документе. В обычной (бумажной) информатике эти проблемы решаются за счет того, что информация в документе и рукописная подпись автора жестко связаны с физическим носителем (бумагой). В электронных документах на машинных носителях такой связи нет.

Целью аутентификации электронных документов является их защита от возможных видов злоумышленных действий, к которым относятся:

- ***активный перехват*** - нарушитель, подключившийся к сети, перехватывает документы (файлы) и изменяет их;
- ***маскарад*** - абонент *C* посылает документ абоненту *B* от имени абонента *A*;
- ***ренегатство*** - абонент *A* заявляет, что не посылал сообщения абоненту *B*, хотя на самом деле послал;

- ***подмена*** - абонент *B* изменяет или формирует новый документ и заявляет, что получил его от абонента *A*;
- ***повтор*** - абонент *C* повторяет ранее переданный документ, который абонент *A* посылал абоненту *B*.

Эти виды злоумышленных действий могут нанести существенный ущерб банковским и коммерческим структурам, государственным предприятиям и организациям, частным лицам, применяющим в своей деятельности компьютерные информационные технологии.

При обработке документов в электронной форме совершенно непригодны традиционные способы установления подлинности по рукописной подписи и оттиску печати на бумажном документе. Принципиально новым решением является электронная цифровая подпись (ЭЦП).

Электронная цифровая подпись используется для аутентификации текстов, передаваемых по телекоммуникационным каналам. Функционально она аналогична обычной рукописной подписи и обладает ее основными достоинствами:

1. удостоверяет, что подписанный текст исходит от лица, поставившего подпись;
2. не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом;
3. гарантирует целостность подписанного текста.

Цифровая подпись представляет собой относительно небольшое количество дополнительной цифровой информации, передаваемой вместе с подписываемым текстом.

Система ЭЦП включает две процедуры:

1. процедуру постановки подписи;

2. процедуру проверки подписи.

В процедуре постановки подписи используется секретный ключ отправителя сообщения, в процедуре проверки подписи - открытый ключ отправителя.

При формировании ЭЦП отправитель прежде всего вычисляет хэш-функцию $\boldsymbol{h(M)}$ подписываемого текста М. Вычисленное значение хэш-функции $\boldsymbol{h(M)}$ представляет собой один короткий блок информации т, характеризующий весь текст *M* в целом. Затем число m шифруется секретным ключом отправителя. Получаемая при этом пара чисел представляет собой ЭЦП для данного текста *M*.

При проверке ЭЦП получатель сообщения снова вычисляет хэш-функцию $\boldsymbol{m = h(M)}$ принятого по каналу текста *M*, после чего при помощи открытого ключа отправителя проверяет, соответствует ли полученная подпись вычисленному значению m хэш-функции.

Принципиальным моментом в системе ЭЦП является невозможность подделки ЭЦП пользователя без знания его секретного ключа подписывания.

В качестве подписываемого документа может быть использован любой файл. Подписанный файл создается из неподписанного путем добавления в него одной или более электронных подписей.

Каждая подпись содержит следующую информацию:

- дату подписи;
- срок окончания действия ключа данной подписи;
- информацию о лице, подписавшем файл (Ф.И.О.. должность. краткое наименование фирмы);
- идентификатор подписавшего (имя открытого ключа);
- собственно цифровую подпись.

Технология применения системы ЭЦП предполагает наличие сети абонентов, посылающих друг другу подписанные электронные документы.

Для каждого абонента генерируется пара ключей: секретный и открытый. Секретный ключ хранится абонентом в тайне и используется им для формирования ЭЦП. Открытый ключ известен всем другим пользователям и предназначен для проверки ЭЦП получателем подписанного электронного документа. Иначе говоря, открытый ключ является необходимым инструментом, позволяющим проверить подлинность электронного документа и автора подписи. Открытый ключ не позволяет вычислить секретный ключ.

Для генерации пары ключей (секретного и открытого) в алгоритмах ЭЦП, как и в асимметричных системах шифрования, используются разные математические схемы, основанные на применении однонаправленных функций. Эти схемы разделяются на две группы. В основе такого разделения лежат известные сложные вычислительные задачи:

- задача факторизации (разложения на множители) больших целых чисел;
- задача дискретного логарифмирования.

11.5 Алгоритм цифровой подписи RSA

Первой и наиболее известной во всем мире конкретной системой ЭЦП стала система RSA, математическая схема которой была разработана в 1977 г. в Массачуссетском технологическом институте США.

Сначала необходимо вычислить пару ключей (секретный ключ и открытый ключ). Для этого отправитель (автор) электронных документов вычисляет два больших простых числа P и Q, затем находит их произведение

$$N = P \cdot Q$$

и значение функции

$$\varphi(N) = (P-1)(Q-1).$$

Далее отправитель вычисляет число ***E*** из условий:

$$E<cp(N),\quad HOA<E,cp(N))=1$$

и число ***D*** из условий:

$$D<N,\quad E*D=1(modcp\{N)).$$

Пара чисел ***(E, N)*** является открытым ключом. Эту пару чисел автор передает партнерам по переписке для проверки его цифровых подписей. Число ***D*** сохраняется автором как секретный ключ для подписывания.

Обобщенная схема формирования и проверки цифровой подписи RSA показана на рис. 11.2.

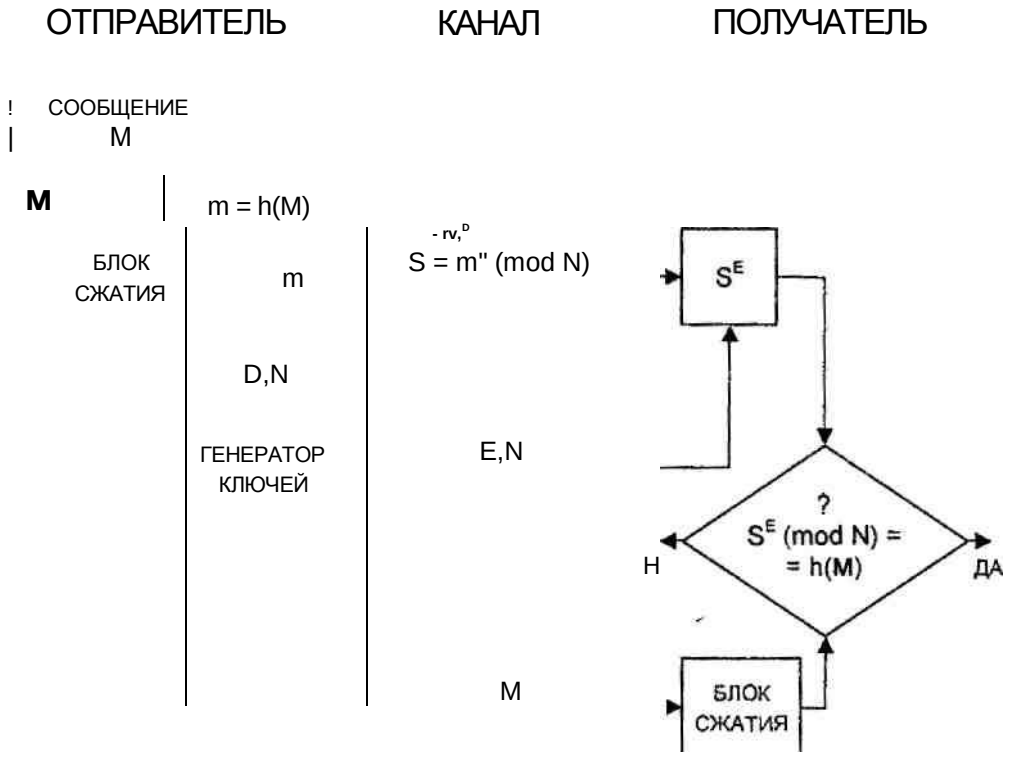

Рис. 11.2 - Обобщенная схема формирования и проверки ЭЦП RSA

Допустим, что отправитель хочет подписать сообщение *M* перед его отправкой. Сначала сообщение *M* (блок информации, файл, таблица) сжимают с помощью хэш-функции $h(\cdot)$ в целое число *m*:

$$\boldsymbol{m = h(M)}.$$

Затем вычисляют цифровую подпись *S* под электронным документом *M*, используя хэш-значение m и секретный ключ ***D***:

$$\boldsymbol{S = m^D \bmod N}.$$

Пара ***(M, S)*** передается партнеру-получателю как электронный документ ***M***, подписанный цифровой подписью ***S***, причем подпись ***S*** сформирована обладателем секретного ключа ***D***,

После приема пары ***(M, S)*** получатель вычисляет хэш-значение сообщения M двумя разными способами. Прежде всего он восстанавливает

хэш-значение m', применяя криптографическое преобразование подписи S с использованием открытого ключа E:

$$m' = S^E \bmod N.$$

Кроме того, он находит результат хэширования принятого сообщения M с помощью такой же хэш-функции $h(\cdot)$:

$$m = h(M).$$

Если соблюдается равенство вычисленных значений, т.е.

$$SE \bmod N = h(M),$$

то получатель признает пару (M, S) подлинной. Доказано, что только обладатель секретного ключа D может сформировать цифровую подпись S по документу M, а определить секретное число D по открытому числу E не легче, чем разложить модуль N на множители.

Кроме того, можно строго математически доказать, что результат проверки цифровой подписи **S** будет положительным только в том случае, если при вычислении S был использован секретный ключ D, соответствующий открытому ключу E. Поэтому открытый ключ E иногда называют «идентификатором» подписавшего.

Недостатки алгоритма цифровой подписи RSA.

- При вычислении модуля N, ключей E и D для системы цифровой подписи RSA необходимо проверять большое количество дополнительных условий, что сделать практически трудно. Невыполнение любого из этих условий делает возможным фальсификацию цифровой подписи со стороны того, кто обнаружит такое невыполнение. При подписании важных документов нельзя допускать такую возможность даже теоретически.
- Для обеспечения криптостойкости цифровой подписи RSA по отношению к попыткам фальсификации на уровне, например, национального стандарта США на шифрование информации (алгоритм DES), т.е. 10 , необходимо использовать при вычислениях N, D и E целые числа не менее 2^{512} (или около 10^{154}) каждое, что требует больших вычислительных затрат, превышающих на 20...30% вычислительные затраты других алгоритмов цифровой подписи при сохранении того же уровня криптостойкости.
- Цифровая подпись RSA уязвима к так называемой мультипликативной атаке. Иначе говоря, алгоритм цифровой подписи RSA позволяет злоумышленнику без знания секретного ключа D сформировать подписи под теми документами, у которых результат хэширования можно вычислить как произведение результатов хэширования уже подписанных документов.