Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра защиты информации

В. Ф. Голиков, А. В. Курилович

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

Учебно-методическое пособие

для студентов специальностей «Сети телекоммуникаций»
и «Защита информации в телекоммуникациях»
всех форм обучения

В 2-х частях

Часть 2

Минск 2008

УДК 621.391.25 (076)
ББК 32.811 я 7
Г 60

Р е ц е н з е н т

зав. кафедрой информационно-измерительной техники и технологий,
д-р физ.-мат. наук И. Е. Зуйков

Голиков, В. Ф.

Г 60 Криптографическая защита информации в телекоммуникационных системах : учеб.-метод. пособие для студ. спец. «Сети телекоммуникаций» и «Защита информации в телекоммуникациях» всех форм обуч. В 2 ч. Ч. 2 / В. Ф. Голиков, А. В. Курилович. – Минск : БГУИР, 2008. – 30 с. : ил.

ISBN 978-985-488-289-5 (ч.2)

Рассмотрены основные методы защиты информации в телекоммуникационных системах: правовые, организационные и технические. Особое внимание уделено техническим методам и средствам защиты информации от несанкционированного доступа, среди которых наиболее эффективными являются криптографические методы.

Пособие предназначено для студентов высших учебных заведений, обучающихся по специальности «Защита информации», а также будет полезно для студентов других специальностей, изучающих дисциплину «Основы защиты информации».

УДК 621.391.25 (076)
ББК 32.811 я 7

Часть 1 настоящего учебно-методического пособия издана в БГУИР в 2006 г.

ISBN 978-985-488-289-5 (ч.2)
ISBN 978-985-488-288-8
ISBN 985-444-987-4

СОДЕРЖАНИЕ

1. ЭЛЕКТРОННАЯ ЦИФРОВАЯ ПОДПИСЬ

1.1. Общие сведения

При обмене сообщениями через телекоммуникационные системы (ТКС) возникает задача подтверждения их подлинности (подтверждения авторства и целостности). Такая же проблема существует и при переходе от юридически значимых бумажных документов к электронным. Сообщения, для которых эта проблема актуальна, будем в дальнейшем называть электронными документами.

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 С посылает документ абоненту В от имени абонента А;
- ренегатство – абонент А заявляет, что не посылал сообщения абоненту В, хотя на самом деле послал;
- подмена – абонент В изменяет или формирует новый документ и заявляет, что получил его от абонента А;
- повтор – абонент С повторяет ранее переданный документ, который абонент А посылал абоненту В.

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.

Естественно, что для электронных документов традиционные способы установления подлинности по рукописной подписи и оттиску печати на бумажном документе совершенно непригодны, поэтому для подтверждения подлинности документа используется специфическая криптографическая процедура, называемая электронной цифровой подписью (ЭЦП).

ЭЦП функционально аналогична обычной рукописной подписи и обладает ее основными достоинствами:

удостоверяет, что подписанный текст исходит от лица, поставившего подпись;

не дает самому этому лицу возможности отказаться от обязательств, связанных с подписанным текстом;

гарантирует целостность подписанного текста.

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

Технология 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 $h(M)$ подписываемого документа M. Вычисленное значение хэш-функции $h(M)$ представляет собой один короткий блок информации m, характеризующий весь документ M в целом. Затем число m «шифруется» секретным ключом отправителя. Получаемая при этом пара чисел представляет собой ЭЦП для данного документа M. В принципе можно обойтись без предварительного хэширования документа, а «шифровать» весь документ, однако в этом случае придется иметь дело с гораздо большим по размеру файлом. Употребление слова «шифровать» здесь весьма условное и справедливо при использовании алгоритма RSA, для других алгоритмов точнее говорить «преобразовывать».

При проверке ЭЦП получатель сообщения снова вычисляет хэш-функцию $m = h(M)$ принятого по каналу документа 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.

1.2. Однонаправленные хэш-функции

Хэш-функция предназначена для сжатия подписываемого документа M до нескольких десятков или сотен бит. Хэш-функция h принимает в качестве аргумента сообщение (документ) M произвольной длины и возвращает хэш-значение $h(M) = H$ фиксированной длины. Обычно хэшированная информация является сжатым двоичным представлением основного сообщения произвольной длины. Следует отметить, что значение хэш-функции $h(M)$ сложным образом зависит от документа M и не позволяет восстановить сам документ M.

Хэш-функция должна удовлетворять целому 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 неразрешима;

вероятность того, что значения хэш-функции двух различных документов (вне зависимости от их длин) совпадут, должна быть ничтожно мала.

Большинство хэш-функций строится на основе однонаправленной функции f, аргументами которой являются две величины: блок исходного документа M_i и хэш-значение H_{i-1} предыдущего блока документа (рис. 1.1): $H_i = f\left(M_i, H_{i-1}\right)$.

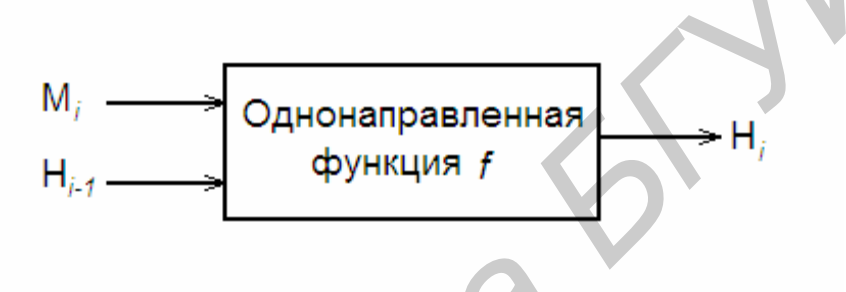

Рис. 1.1. Общая схема вычисления однонаправленной хэш-функции

Хэш-значение, вычисляемое при вводе последнего блока текста, становится хэш-значением всего сообщения M. В результате однонаправленная хэш-функция всегда формирует выход фиксированной длины n (независимо от длины входного текста).

Часто функции хэширования строят, используя в качестве однонаправленной функции симметричный блочный алгоритм шифрования (DES, ГОСТ 28147-89) в режиме с обратной связью, принимая последний блок шифротекста за хэш-значение всего документа. Так как длина блока в указанных алго-

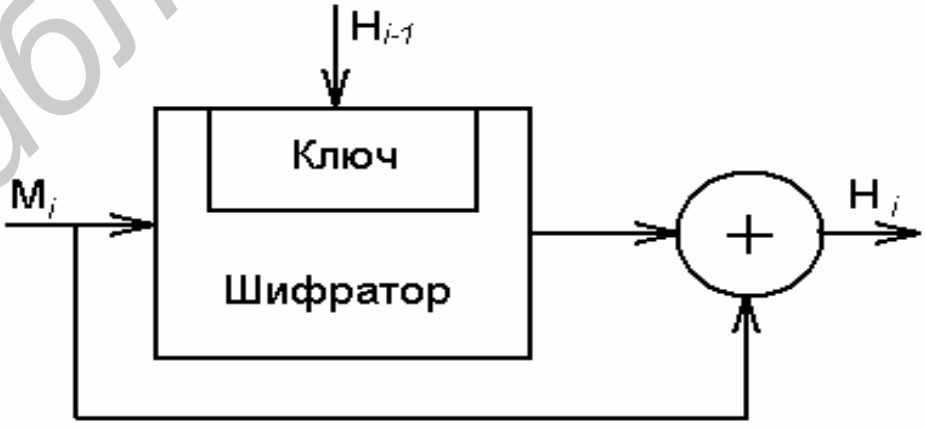

Рис. 1.2. Схема вычисления однонаправленной функции хэширования на базе блочного алгоритма шифрования

ритмах невелика (64 бита), то часто в качестве хэш-значения используют два блока шифротекста. Одна из возможных схем хэширования на основе блочного алгоритма шифрования изображена на рис. 1.2.

1.3. Алгоритм электронной 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 Массачусетском 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 Эйлера $j(n) = (P-1)\cdot(Q-1)$. Далее отправитель вычисляет число K_O из условий: $K_O \le j(n)$, НОД$(K_O, j(n)) = 1$ и число K_C из условий: $K_C < n$, $K_O \cdot K_C \equiv 1 \bmod(j(n))$.

Пара чисел (K_O, n) является открытым ключом. Эту пару чисел автор передает партнерам по переписке для проверки его цифровых подписей. Число K_C сохраняется автором как секретный ключ для подписывания. Обобщенная схема формирования и проверки цифровой подписи RSA показана на рис. 1.3.

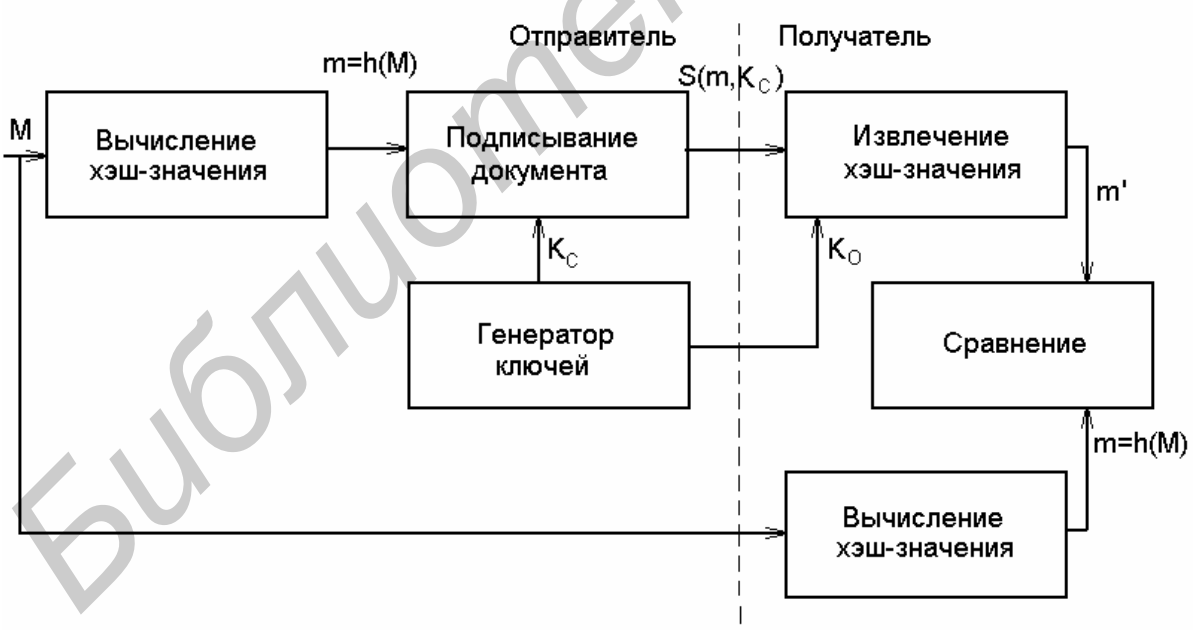

Рис. 1.3. Обобщенная схема алгоритма ЭЦП RSA

Допустим, что отправитель хочет подписать сообщение M перед его отправкой. Сначала сообщение M (блок информации, файл, таблица) сжимают с помощью хэш-функции h в целое число $m = h(M)$. Затем вычисляют цифровую подпись S под электронным документом M, используя хэш-

значение m и секретный ключ K_C: $S = m^{K_C} \bmod n$. Пара (M, S) передается партнеру-получателю как электронный документ M, подписанный цифровой подписью S, причем подпись S сформирована обладателем секретного ключа K_C. 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 K_O: $m' = S^{K_O} \bmod n$. Кроме того, он находит результат хэширования принятого сообщения M с помощью такой же хэш-функции h: $m = h(M)$. Если соблюдается равенство вычисленных значений, т.е. $S^{K_O} \bmod n = h(M)$, то получатель признает пару (M, S) подлинной. Доказано, что только обладатель секретного ключа K_C может сформировать цифровую подпись S по документу M, а определить секретное число K_C по открытому числу K_O 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 K_C, соответствующий открытому ключу K_O. Поэтому открытый ключ K_O иногда называют «идентификатором» подписавшего.

Алгоритм цифровой подписи RSA имеет ряд недостатков.

1. При вычислении модуля n, ключей K_C и K_O для системы цифровой подписи RSA необходимо проверять большое количество дополнительных условий, что сделать практически трудно. Невыполнение любого из этих условий делает возможной фальсификацию цифровой подписи. При подписании важных документов нельзя допускать такую возможность даже теоретически.

2. 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^{18}, необходимо использовать при вычислениях n, K_C и K_O 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.

3. Цифровая подпись RSA уязвима к так называемой мультипликативной атаке. Иначе говоря, алгоритм цифровой подписи RSA позволяет злоумышленнику без знания секретного ключа K_C сформировать подписи под теми документами, у которых результат хэширования можно вычислить как произведение результатов хэширования уже подписанных документов.

Пример 1.1. Допустим, что злоумышленник может сконструировать три сообщения M_1, M_2 и M_3, у которых хэш-значения $m_1 = h(M_1)$,

$m_2 = h(M_2)$, $m_3 = h(M_3)$, причем $m_3 = m_1 \cdot m_2 (\mathrm{mod}\, n)$. Допустим также, что для двух сообщений M_1 и M_2 получены законные подписи $S_1 = m_1^{K_C} (\mathrm{mod}\, n)$ и $S_2 = m_2^{K_C} (\mathrm{mod}\, n)$. Тогда злоумышленник может легко вычислить подпись S_3 для документа M_3, даже не зная секретного ключа $S_3 = S_1 \cdot S_2 (\mathrm{mod}\, n)$. Действительно

$$S_1 \cdot S_2 (\mathrm{mod}\, n) = m_1^{K_C} \cdot m_1^{K_C} (\mathrm{mod}\, n) = \left(m_1 \cdot m_2\right)^{K_C} (\mathrm{mod}\, n) = m_3^{K_C} (\mathrm{mod}\, n).$$

1.4. Алгоритм цифровой подписи Эль Гамаля (EGSA)

Название EGSA происходит от слов El Gamal Signature Algorithm (алгоритм цифровой подписи Эль Гамаля). Идея EGSA основана на том, что для обоснования практической невозможности фальсификации цифровой подписи может быть использована более сложная вычислительная задача, чем разложение на множители большого целого числа – задача дискретного логарифмирования. Кроме того, Эль Гамалю удалось избежать явной слабости алгоритма цифровой подписи RSA, связанной с возможностью подделки цифровой подписи под некоторыми сообщениями без определения секретного ключа. Для генерации пары ключей (открытый ключ – секретный ключ) сначала выбирают некоторое большое простое целое число P и большое целое число G, причем $G < P$. Отправитель и получатель подписанного документа используют при вычислениях одинаковые большие целые числа P ($: 10^{308}$ или $: 2^{1024}$) и G ($: 10^{154}$ или $: 2^{512}$), которые не являются секретными. Отправитель выбирает случайное целое число K_C, $1 < K_C < P-1$, и вычисляет $K_O = G^{K_C} \bmod P$. Число K_O является открытым ключом, используемым для проверки подписи отправителя. Число K_O открыто передается всем потенциальным получателям документов. Число K_C является секретным ключом отправителя для подписывания документов и должно храниться в секрете. Для того чтобы подписать сообщение M, сначала отправитель хэширует его с помощью хэш-функции $h(*)$ в целое число $m = h(M)$, $1 < m < (P-1)$, и генерирует случайное целое число K, $1 < K < (P-1)$, такое, что K и $(P-1)$ являются взаимно простыми. Затем отправитель вычисляет целое число a: $a = G^K \bmod P$ и, применяя расширенный алгоритм Евклида, вычисляет с помощью секретного ключа K_C целое число b из уравнения $m = K_C \cdot a + K \cdot b(\mathrm{mod}(P-1))$.

Пара чисел (a,b) образует цифровую подпись S: $S = (a,b)$, проставляемую под документом M. Тройка чисел (M,a,b) передается получателю, в то время как пара чисел (K_C, K) держится в секрете.

После приема подписанного сообщения (M, a, b) получатель должен проверить, соответствует ли подпись $S = (a, b)$ сообщению M. Для этого получатель сначала вычисляет по принятому сообщению M число $m = h(M)$, т.е. хэширует принятое сообщение M. Затем получатель вычисляет значение $A = K_O^a \cdot a^b \pmod P$ и признает сообщение M подлинным только в том случае, если $A = G^m \bmod P$. Иначе говоря, получатель проверяет справедливость соотношения $K_O^a \cdot a^b \pmod P = G^m \pmod P$.

Можно строго математически доказать, что последнее равенство будет выполняться тогда, и только тогда, когда подпись $S = (a, b)$ под документом M получена с помощью именно того секретного ключа K_C, из которого был получен открытый ключ K_O. Таким образом, можно надежно удостовериться, что отправителем сообщения M был обладатель именно данного секретного ключа K_C, не раскрывая при этом сам ключ, и что отправитель подписал именно этот конкретный документ M.

Следует отметить, что выполнение каждой подписи по методу Эль Гамаля требует нового значения K, причем это значение должно выбираться случайным образом. Если нарушитель раскроет когда-либо значение K, повторно используемое отправителем, то он сможет раскрыть секретный ключ K_C отправителя. Следует отметить, что схема Эль Гамаля является характерным примером подхода, который допускает пересылку сообщения M в открытой форме вместе с присоединенным аутентификатором (a, b). В таких случаях процедура установления подлинности принятого сообщения состоит в проверке соответствия аутентификатора сообщению.

Схема цифровой подписи Эль Гамаля имеет ряд преимуществ по сравнению со схемой цифровой подписи RSA:

1. При заданном уровне стойкости алгоритма цифровой подписи целые числа, участвующие в вычислениях, имеют запись на 25 % короче, что уменьшает сложность вычислений почти в два раза и позволяет заметно сократить объем используемой памяти.

2. При выборе модуля P достаточно проверить, является ли это число простым и имеется ли у числа $(P-1)$ большой простой множитель (т.е. всего два достаточно просто проверяемых условия).

3. Процедура формирования подписи по схеме Эль Гамаля не позволяет вычислять цифровые подписи под новыми сообщениями без знания секретного ключа (как в RSA).

Однако алгоритм цифровой подписи Эль Гамаля имеет и некоторые недостатки по сравнению со схемой подписи RSA. В частности, длина цифровой подписи получается в 1,5 раза больше, что, в свою очередь, увеличивает время ее вычисления.

Пример 1.2. Выберем числа $P=11$, $G=2$ и секретный ключ $K_C=8$. Вычислим значение открытого ключа $K_O=G^{K_C}\bmod P=2^8\bmod 11=3$. Предположим, что исходное сообщение M характеризуется хэш-значением $m=5$. Для того чтобы вычислить цифровую подпись для сообщения M, имеющего хэш-значение $m=5$, сначала выберем случайное целое число $K=9$. Убедимся, что числа K и $(P-1)$ являются взаимно простыми. Действительно, НОД$(9,10)=1$. Далее вычислим элементы a и b подписи: $a=G^K\bmod P=2^9\bmod 11=6$, элемент b определяем из уравнения $m=K_C\cdot a+K\cdot b(\bmod(P-1))$, используя расширенный алгоритм Евклида. При $m=5$, $a=6$, $K_C=8$, $P=11$ получаем $5=6\cdot 8+9\cdot b(\bmod 10)$ или $9\cdot b=-43(\bmod 10)$. Решением является $b=3$. Цифровая подпись представляет собой пару $a=6$, $b=3$. Далее отправитель передает подписанное сообщение. Приняв подписанное сообщение и открытый ключ $K_O=3$, получатель вычисляет хэш-значение для сообщения M: $m=5$, а затем вычисляет:

1) $K_O^a\cdot a^b(\bmod P)=3^6\cdot 6^3(\bmod 11)=10$;

2) $G^m\bmod P=2^5\bmod 11=10$.

Так как эти два целых числа равны, принятое получателем сообщение признается подлинным.

1.5. Белорусские стандарты ЭЦП и функции хэширования

Белорусские стандарты, регламентирующие использование электронной цифровой подписи, официальное название которых «Процедура выработки и проверки ЭЦП» и «Функция хэширования», были разработаны группой белорусских специалистов в 1999 г. и официально приняты в 2000 г.

В этих стандартах наряду с элементами классических процедур ЭЦП используются современные идеи, позволяющие увеличить криптостойкость и быстродействие. Так, открытый ключ и секретный ключ связаны известным соотношением: $K_O=\left(a^{K_C}\right)\bmod P$, которое позволяет легко вычислить K_O по K_C, но очень сложным является решение обратной задачи – вычисления K_C по K_O. К подписываемому сообщению добавляется случайная компонента t, что усложняет возможный подбор хэш-значения злоумышленником по известному тексту сообщения.

Обозначения, принятые в стандарте СТБ-1176.02-99

- B_p – множество, состоящее из чисел $1,2,\mathbf{K},p-1$;
- $c:=d$ – присвоение параметру c значения d;

- $c \bmod d$ – остаток от деления c на d, где c – натуральное число или ноль, d – натуральное число;
- $c^{-1} \bmod d$ – натуральное число b такое, что $b < d$ и $(cb) \bmod d = 1$, где c и d – взаимно простые числа;
- $\lceil c \rceil$ – наименьшее целое число, не меньшее чем c;
- $\lfloor c \rfloor$ – наибольшее целое число, не большее чем c;
- $c = \sum_{i=0}^{k-1} c_i (2^b)^i$ – разложение неотрицательного целого числа c по основанию 2^b, где k и b – натуральные числа,
- c_i – целое число, $0 \le c_i < 2^b$;
- $\oplus$ – бинарная операция, определенная на множестве неотрицательных целых чисел по формуле $d \oplus b = \sum_{i=0}^{k-1} \left((d_i + b_i) \bmod 2 \right) 2^i$, где $d = \sum_{i=0}^{k-1} d_i 2^i$, $b = \sum_{i=0}^{k-1} b_i 2^i$, $d_0, \mathbf{K}, d_{k-1}, b_0, \mathbf{K}, b_{k-1} \in \{0,1\}$;
- $\mathbf{o}$ – операция $\mathbf{o} : B_p \times B_p \to B_p$ определяется для любых $c \in B_p$ и $d \in B_p$ по формуле $c \mathbf{o} d = (cd(2^{l+2})^{-1}) \bmod p$;
- $c^{(k)}$ – степень числа на основе операции $\mathbf{o}$, определяется индуктивно по формуле $c^{(k)} = \begin{cases} c, & k = 1, \\ c^{(k-1)} \mathbf{o}\, c, & k > 1, \end{cases}$ где k – натуральное число;
- h – функция хэширования, процедура вычисления значений которой соответствует СТБ.

Процедура выработки ЭЦП

1. Выбираются параметры l и r, которые определяют уровень криптографической стойкости ЭЦП. Число l является длиной записи числа p в системе счисления по основанию 2, r является длиной записи числа q в системе счисления по основанию 2.

2. В соответствии с выбранными l и r генерируются простые числа p и q, такие, что q делит $p-1$ нацело.

3. Генерируется случайное число d, $0 < d < p$.

4. Вычисляется $a = d^{\left(\frac{p-1}{q}\right)}$. Если $a \equiv 2^{l+2} \bmod p$, то перейти к п. 3.

5. Генерируется случайное число x, $0 < x < q$, которое является секретным ключом.

6. Вычисляется число $y = a^{(x)}$, которое является открытым ключом.

7. Генерируется случайное число k, $0 < k < q$.

8. Вычисляется $t = a^{(k)}$. Далее производится разложение числа t по основанию 2^8, т.е. $t = \sum_{i=0}^{n-1} t_i (2^8)^i$. Таким образом, получаются коэффициенты $t_0, t_1, \mathbf{K}, t_{n-1}$.

9. Формируется последовательность $M_t = (t_0, t_1, \mathbf{K}, t_{n-1}, m_1, m_2, \mathbf{K}, m_z)$, состоящая из коэффициентов $t_0, t_1, \mathbf{K}, t_{n-1}$ и блоков открытого текста $m_1, m_2, \mathbf{K}, m_z$.

10. Вычисляется значение хэш-функции $U = h(M_t)$. Если $U = 0$, то перейти к п. 6.

11. Вычисляется $V = (k - xU) \bmod q$. Если $V = 0$, то перейти к п. 6.

12. Вычисляется $S = U \cdot 2^r + V$. ЭЦП последовательности M_t есть число S.

13. Отправляется M_t, S.

Процедура проверки ЭЦП

1. Вычисляется $V = S \bmod 2^r$.

2. Вычисляется $U = (S - V)/2^r$.

3. Если хотя бы одно из условий $0 < U < 2^r$ и $0 < V < q$ не выполнено, то ЭЦП считается недействительной и работа алгоритма завершается.

4. Вычисляется $t' = a^{(V)} \mathbf{o} y^{(U)}$.

5. Число t' разлагается по основанию 2^8, т.е. $t' = \sum_{i=0}^{n-1} t'_i (2^8)^i$. Таким образом, получаются коэффициенты $t'_0, t'_1, \mathbf{K}, t'_{n-1}$.

6. Формируется последовательность $M'_t = (t'_0, t'_1, \mathbf{K}, t'_{n-1}, m_1, m_2, \mathbf{K}, m_z)$, состоящая из коэффициентов $t'_0, t'_1, \mathbf{K}, t'_{n-1}$ и блоков открытого текста $m_1, m_2, \mathbf{K}, m_z$.

7. Вычисляется хэш-функция $W = h(M'_t)$.

8. Проверяется условие $W = U$. При совпадении W и U принимается решение о том, что ЭЦП была создана при помощи личного ключа подписи x, связанного с открытым ключом проверки подписи y, а также ЭЦП и последовательность M_t не были изменены с момента их создания. В противном случае подпись считается недействительной.

Стандарт «Процедура выработки и проверки ЭЦП» содержит алгоритмы и процедуры выработки и проверки электронной цифровой подписи, а также подробные инструкции по:

- выбору величин r и l (размер p и q);

- генерации p и q;
- генерации a.

2. АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЕЙ В ТКС

2.1. Общие сведения

Аутентификация означает установление подлинности и обеспечивает работу в сети только санкционированных пользователей. Чаще всего аутентификация проводится при входе в сеть, но может проводиться и во время работы. Обычно она следует после процесса идентификации, во время которого пользователь сообщает свой идентификатор (называет себя). В процедуре аутентификации участвуют две стороны: пользователь доказывает свою подлинность, а сеть проверяет это доказательство и принимает решение. В качестве доказательства используют:

- знание секрета (пароля);
- владение уникальным предметом (физическим ключом);
- предъявление биометрической характеристики (отпечаток пальца, рисунок радужной оболочки глаза, голос).

Наиболее распространенное средство аутентификации – пароль. Используется как при входе в систему, так и в процессе работы. Пароль может вводиться с клавиатуры, или с различных носителей цифровой информации, или комбинированно. При использовании паролей необходимо соблюдать обязательные требования: по правилам генерации (длина, случайность символов), хранения (хранить в защищенном месте), использования (в зашифрованном виде), отзыва.

В качестве субъектов аутентификации могут выступать не только пользователи, но и различные процессы или устройства. Причем процесс аутентификации может носить обоюдный характер. Обе стороны должны доказать свою подлинность. Например, пользователь, обращающийся к корпоративному серверу, должен убедится, что имеет дело с сервером своего предприятия. В этом случае процедура называется взаимной аутентификацией.

2.2. Удаленная аутентификация пользователей с использованием пароля

Процесс удаленной аутентификации обычно выполняют в начале сеанса связи. Рассмотрим аутентификацию с использованием пароля.

Пусть стороны А и В знают друг друга и имеют одинаковый секретный ключ K_{AB}. Пользователь А вводит свой идентификатор (PIN). Его программа, используя ключ и PIN, вырабатывает пароль *P*, который вместе с PIN передается по сети к пользователю В. Пользователь В по PIN находит в своей базе

ключ K_{AB}, с помощью которого вырабатывает P, после чего сравнивает значение полученного пароля и выработанного. Схема описанной аутентификации приведена на рис. 2.1.

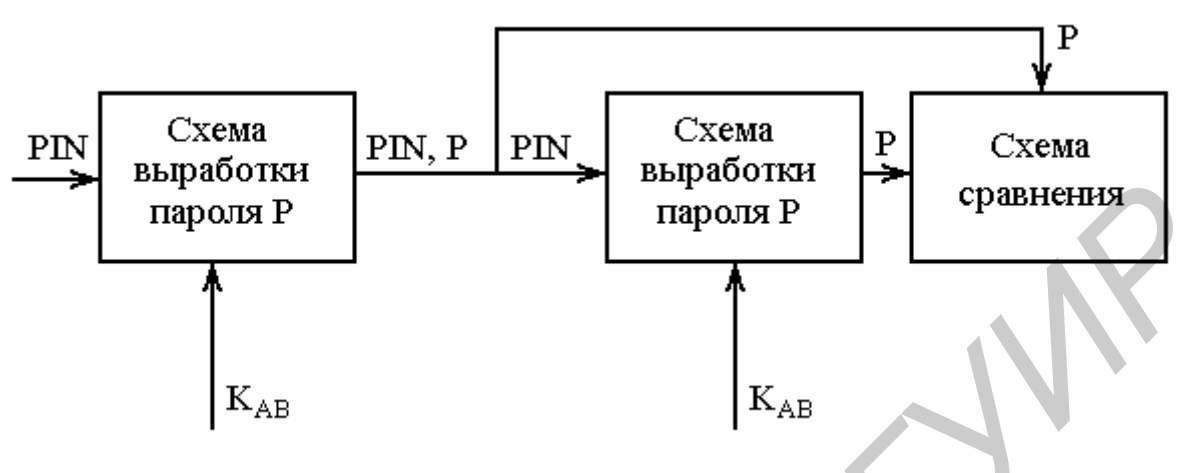

Рис. 2.1. Схема аутентификации с использованием пароля

Рассмотренная схема имеет существенный недостаток: злоумышленник может перехватить пароль P и PIN и позднее использовать их для своей аутентификации. Для устранения этого недостатка используют механизм отметки времени («временной штемпель»). Суть этого механизма заключается в том, что при выработке пароля наряду с ключом используется текущее время в виде некоторого интервала, в пределах которого пароль действителен, аналогично вырабатывается пароль на стороне B, в этом случае устаревшим паролем нельзя воспользоваться.

2.3. Удаленная аутентификация пользователей с использованием механизма запроса–ответа

Процедура состоит в следующем. Если пользователь A хочет быть уверенным, что сообщения, получаемые им от пользователя B, не являются ложными, он включает в посылаемое для B сообщение непредсказуемый элемент – запрос X (например некоторое случайное число). При ответе пользователь B должен выполнить над этим элементом некоторую операцию (например вычислить некоторую функцию $f(X)$). Это невозможно осуществить заранее, так как пользователю B неизвестно, какое случайное число X придет в запросе. Получив ответ с результатом действий B, пользователь A может быть уверен, что B – подлинный. Недостаток этого метода – возможность установления закономерности между запросом и ответом.

Механизм запрос–ответ используется в более сложной процедуре аутентификации – «рукопожатии».

Процедура «рукопожатия» базируется на указанном выше механизме и заключается во взаимной проверке ключей, используемых сторонами. Иначе

говоря, стороны признают друг друга законными партнерами, если докажут друг другу, что обладают правильными ключами. Процедуру «рукопожатия» обычно применяют в компьютерных сетях при организации сеанса связи между пользователями, пользователем и хост-компьютером, между хост-компьютерами и т.д.

Рассмотрим в качестве примера процедуру рукопожатия для двух пользователей А и В. (Это допущение не влияет на общность рассмотрения. Такая же процедура используется, когда вступающие в связь стороны не являются пользователями.) Пусть применяется симметричная криптосистема. Пользователи А и В разделяют один и тот же секретный ключ K_{AB}. Вся процедура показана на рис. 2.2.

Пусть пользователь А инициирует процедуру рукопожатия, отправляя пользователю В свой идентификатор ID_{A} в открытой форме. Пользователь В, получив идентификатор ID_{A}, находит в базе данных секретный ключ K_{AB} и вводит его в свою криптосистему.

Тем временем пользователь А генерирует случайную последовательность S с помощью псевдослучайного генератора PG и отправляет ее пользователю В в виде криптограммы $E_{K_{\text{AB}}}(S)$. Пользователь В расшифровывает эту криптограмму и раскрывает исходный вид последовательности S. Затем оба пользователя А и В преобразуют последовательность S, используя открытую одностороннюю функцию $a(*)$.

Пользователь В шифрует сообщение $a(S)$ и отправляет эту криптограмму пользователю А. Наконец, пользователь А расшифровывает эту криптограмму и сравнивает полученное сообщение $a'(S)$ с исходным $a(S)$. Если эти сообщения равны, пользователь А признает подлинность пользователя В.

Очевидно, пользователь В проверяет подлинность пользователя А таким же способом. Обе эти процедуры образуют процедуру рукопожатия, которая обычно выполняется в самом начале любого сеанса связи между любыми двумя сторонами в компьютерных сетях.

Достоинством модели рукопожатия является то, что ни один из участников сеанса связи не получает никакой секретной информации во время процедуры подтверждения подлинности.

Процедура рукопожатия была рассмотрена в предположении, что пользователи А и В доверяют друг другу и имеют общий *секретный сеансовый ключ*. Однако нередки ситуации когда пользователи должны осуществить взаимную аутентификацию, не доверяя друг другу и не обмениваясь никакой конфиденциальной информацией.

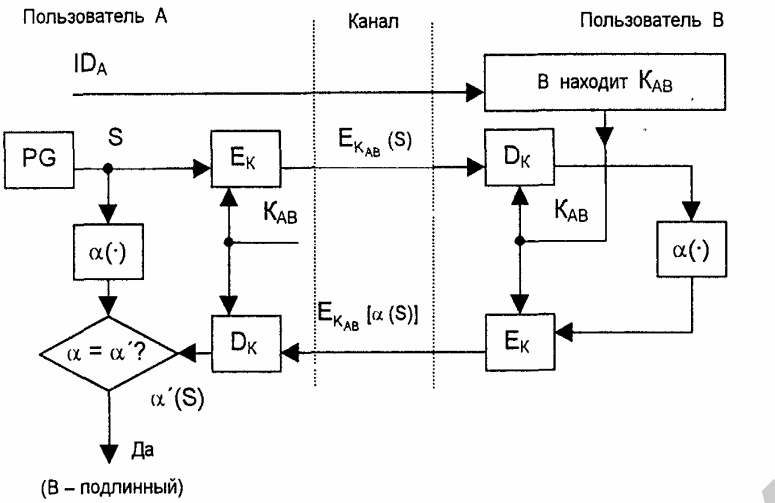

Рис. 2.2. Схема процедуры рукопожатия (пользователь А проверяет подлинность пользователя В)

2.4. Протоколы идентификации с нулевой передачей знаний

Описанная выше ситуация характерна при использовании интеллектуальных карт (смарт-карт) для разнообразных коммерческих, гражданских и военных применений (кредитные карты, карты социального страхования, карты доступа в охраняемое помещение, компьютерные пароли и ключи, и т.п.). Во многих приложениях главная проблема заключается в том, чтобы при предъявлении интеллектуальной карты оперативно обнаружить обман и отказать обманщику в допуске, ответе или обслуживании.

Для безопасного использования интеллектуальных карт разработаны протоколы идентификации с нулевой передачей знаний. Секретный ключ владельца карты становится неотъемлемым признаком его личности. Доказательство знания этого секретного ключа с нулевой передачей этого знания служит доказательством подлинности личности владельца карты.

Упрощенная схема идентификации с нулевой передачей знаний

Схему идентификации с нулевой передачей знаний предложили в 1986 г. У. Фейге, А. Фиат и А. Шамир. Она является наиболее известным доказательством идентичности с нулевой передачей конфиденциальной информации.

Рассмотрим упрощенный вариант схемы идентификации с нулевой передачей знаний для более четкого выявления ее основной концепции. Прежде всего выбирают случайное значение модуля n, который является произведением двух больших простых чисел. Модуль n должен иметь длину 512–1024 бит. Это значение n может быть представлено группе пользователей,

которым придется доказывать свою подлинность. В процессе идентификации участвуют две стороны:

- сторона А, доказывающая свою подлинность,
- сторона В, проверяющая представляемое стороной А доказательство.

Для того чтобы сгенерировать открытый и секретный ключи для стороны А, доверенный арбитр (Центр) выбирает некоторое число V, которое является квадратичным вычетом по модулю n. Иначе говоря, выбирается такое число V, что сравнение $x^2 \equiv V \bmod n$ имеет решение и существует целое число $V^{-1} \bmod n$.

Выбранное значение V является *открытым ключом* для А. Затем вычисляют наименьшее значение S, для которого $S = sqrt\left(V^{-1}\right)\left(\bmod n\right)$. Это значение S является секретным ключом для А.

Теперь можно приступить к выполнению протокола идентификации.

1. Сторона А выбирает некоторое случайное число r, где $r < n$. Затем она вычисляет $x = r^2 \bmod n$ и отправляет x стороне В.

2. Сторона В посылает А случайный бит b.

3. Если $b = 0$, тогда А отправляет r стороне В. Если $b = 1$, то А отправляет стороне В $y = r \cdot S \bmod n$.

4. Если $b = 0$, то сторона В проверяет, что $x = r^2 \bmod n$, чтобы убедиться, что А знает $sqrt\left(x\right)$. Если $b = 1$, сторона В проверяет, что $x = y^2 \cdot V \bmod n$, чтобы быть уверенной, что А знает $sqrt\left(V^{-1}\right)$.

Эти шаги образуют один цикл протокола, называемый аккредитацией. Стороны А и В повторяют этот цикл t раз при разных случайных значениях r и b до тех пор, пока В не убедится, что А знает значение S.

Если сторона А не знает значения S, она может выбрать такое значение r, которое позволит ей обмануть сторону В, если В отправит ей $b = 0$, либо А может выбрать такое r, которое позволит обмануть В, если В отправит ей $b = 1$. Но это невозможно сделать в обоих случаях. Вероятность того, что А обманет В в одном цикле, составляет $1/2$. Вероятность обмануть В в t циклах равна $(1/2)^t$.

Для того чтобы этот протокол работал, сторона А никогда не должна повторно использовать значение r. Если А поступила бы таким образом, а сторона В отправила бы стороне А на шаге 2 другой случайный бит b, то В имела бы оба ответа А. После этого В может вычислить значение S, и для А все закончено.

Параллельная схема идентификации с нулевой передачей знаний

Параллельная схема идентификации позволяет увеличить число аккредитаций, выполняемых за один цикл, и тем самым уменьшить длительность процесса идентификации.

Как и в предыдущем случае, сначала генерируется число n как произведение двух больших чисел. Для того чтобы сгенерировать открытый и секретный ключи для стороны А, сначала выбирают K различных чисел $V_1, V_2, ...V_K$, где каждое V_i является квадратичным вычетом по модулю n. Иначе говоря, выбирают значение V_i таким, что сравнение $x^2 \equiv V_i \bmod n$ имеет решение и существует $V_i^{-1} \bmod n$. Полученная строка $V_1, V_2, ...V_K$ является открытым ключом. Затем вычисляют такие наименьшие значения S_i, что $S_i = sqrt\left(V_i^{-1}\right) \bmod n$. Эта строка $S_1, S_2, ...S_K$ является секретным ключом стороны А.

Процесс идентификации имеет следующий вид:

1. Сторона А выбирает некоторое случайное число r, где $r < n$. Затем она вычисляет $x = r^2 \bmod n$ и посылает x стороне В.

2. Сторона В отправляет стороне А некоторую случайную двоичную строку из K бит: $b_1, b_2, ...b_K$.

3. Сторона А вычисляет $y = r \cdot \left(S^{b_1} \cdot S^{b_2} \cdot ... \cdot S^{b_K}\right) \bmod n$.

Перемножаются только те значения S_i, для которых $b_i = 1$. Например, если $b_i = 1$, то сомножитель S_i входит в произведение, если же $b_i = 0$, то S_i не входит в произведение, и т.д. Вычисленное значение y отправляется стороне В.

4. Сторона В проверяет, что $x = y^2 \cdot \left(V_1^{b_1} \cdot V_2^{b_2} \cdot ... \cdot V_K^{b_K}\right) \bmod n$.

Фактически сторона В перемножает только те значения V_i, для которых $b_i = 1$. Стороны А и В повторяют этот протокол t раз, пока В не убедится, что А знает $S_1, S_2, ...S_K$.

Вероятность того, что А может обмануть В, равна $(1/2)^{Kt}$. Рекомендуется в качестве контрольного значения брать вероятность обмана В равной $(1/2)^{20}$ при $K = 5$ и $t = 4$.

Стороны А и В повторяют этот протокол t раз, каждый раз с разным случайным числом r, пока сторона В не будет удовлетворена.

При малых значениях величин, как в данном примере, не достигается настоящей безопасности. Но если n представляет собой число длиной 512 бит и более, сторона В не сможет узнать ничего о секретном ключе стороны А, кроме того факта, что сторона А знает этот ключ.

3. УПРАВЛЕНИЕ КРИПТОГРАФИЧЕСКИМИ КЛЮЧАМИ

Любая криптографическая система основана на использовании криптографических ключей. В симметричной криптосистеме отправитель и получатель сообщения используют один и тот же секретный ключ. Этот ключ должен быть неизвестен всем остальным и должен периодически обновляться одновременно у отправителя и получателя. Процесс распределения (рассылки) секретных ключей между участниками информационного обмена в симметричных криптосистемах имеет весьма сложный характер.

Асимметричная криптосистема предполагает использование двух ключей – открытого и личного (секретного). Открытый ключ можно разглашать, а личный надо хранить в тайне. При обмене сообщениями необходимо пересылать только открытый ключ. Важным требованием является обеспечение подлинности отправителя сообщения. Это достигается путем взаимной аутентификации участников информационного обмена.

Под *ключевой информацией* понимают совокупность всех действующих в системе ключей. Если не обеспечено достаточно надежное управление ключевой информацией, то, завладев ею, злоумышленник получает неограниченный доступ ко всей информации.

Управление ключами – информационный процесс, включающий реализацию следующих основных функций:

- генерация ключей,
- хранение ключей,
- распределение ключей.

3.1. Генерация ключей

Безопасность любого криптографического алгоритма определяется используемым криптографическим ключом. Добротные криптографические ключи должны иметь достаточную длину и случайные значения бит.

Для получения ключей используются аппаратные и программные средства генерации случайных значений ключей. Как правило, применяют датчики псевдослучайных чисел (ПСЧ). Однако степень случайности генерации чисел должна быть достаточно высокой. Идеальными генераторами являются устройства на основе «натуральных» случайных процессов, например на основе *белого радиошума.*

Если ключ не меняется регулярно, это может привести к его раскрытию и утечке информации. Регулярную замену ключа можно осуществить, используя процедуру модификации ключа.

Модификация ключа – это генерирование нового ключа из предыдущего значения ключа с помощью односторонней (однонаправленной) функции. Участники информационного обмена разделяют один и тот же ключ и одновременно вводят его значение в качестве аргумента в одностороннюю функ-

цию, получая один и тот же результат. Затем они берут определенные биты из этих результатов, чтобы создать новое значение ключа.

Процедура модификации ключа работоспособна, но надо помнить, что новый ключ безопасен в той же мере, в какой был безопасен прежний ключ. Если злоумышленник сможет добыть прежний ключ, то он сможет выполнить процедуру модификации ключа.

Генерация ключей для асимметричных криптосистем с открытыми ключами много сложнее, потому что эти ключи должны обладать определенными математическими свойствами (они должны быть очень большими и простыми и т.д.).

3.2. Хранение ключей

Под функцией хранения ключей понимают организацию их безопасного хранения, учета и удаления.

Секретные ключи никогда не должны записываться в явном виде на носителе, который может быть считан или скопирован. Любая информация об используемых ключах должна быть защищена, в частности, храниться в зашифрованном виде.

Необходимость в хранении и передаче ключей, зашифрованных с помощью других ключей, приводит к концепции иерархии ключей. Суть концепции состоит в том, что вводится иерархия ключей: главный ключ (ГК), ключ шифрования ключей (КК), ключ шифрования данных (КД).

Иерархия ключей может быть:

- двухуровневой (КК/КД),
- трехуровневой (ГК/КК/КД).

Самым нижним уровнем являются рабочие или сеансовые КД, которые используются для шифрования данных, персональных идентификационных номеров (PIN) и аутентификации сообщений. Когда эти ключи надо зашифровать с целью защиты при передаче или хранении, используют ключи следующего уровня – ключи шифрования ключей. Ключи шифрования ключей никогда не должны использоваться как сеансовые (рабочие) КД и наоборот.

Такое разделение функций необходимо для обеспечения максимальной безопасности. Фактически концепция устанавливает, что различные типы рабочих ключей (например, для шифрования данных, аутентификации и т.д.) должны всегда шифроваться с помощью различных версий ключей шифрования ключей.

В частности, ключи шифрования ключей, используемые для пересылки ключей между двумя узлами сети, известны также как ключи обмена между узлами сети. Обычно в канале используются два ключа для обмена между узлами сети, по одному в каждом направлении. Поэтому каждый узел сети будет иметь ключ отправления для обмена с узлами сети и ключ получения для каждого канала, поддерживаемого другим узлом сети.

На верхнем уровне иерархии ключей располагается главный ключ, мастер-ключ. Этот ключ применяют для шифрования КК, когда требуется сохранить их на диске. Обычно в каждом компьютере используется только один мастер-ключ.

Мастер-ключ распространяется между участниками обмена неэлектронным способом при личном контакте, чтобы исключить его перехват и/или компрометацию. Раскрытие противником значения мастер-ключа полностью уничтожает защиту компьютера.

Значение мастер-ключа фиксируется на длительное время (до нескольких недель или месяцев) Поэтому генерация и хранение мастер-ключей являются критическими вопросами криптографической защиты. На практике мастер-ключ компьютера создается случайным выбором из всех возможных значений ключей. Мастер-ключ помещают в защищенный от считывания, записи и механических воздействий блок криптографической системы таким образом, чтобы раскрыть значение этого ключа было невозможно. Однако все же должен существовать способ проверки, является ли значение ключа правильным.

Проблема аутентификации мастер-ключа может быть решена различными путями. Один из способов показан на рис. 3.1.

Администратор, получив новое значение мастер-ключа K_H хост-компьютера, шифрует некоторое сообщение M ключом K_H. Пара (криптограмма $E_{K_H}(M)$, сообщение M) помещается в память компьютера. Всякий раз, когда требуется аутентификация мастер-ключа хост-компьютера, берется сообщение M из памяти и подается в криптографическую систему. Получаемая криптограмма сравнивается с криптограммой, хранящейся в памяти. Если они совпадают, считается, что данный ключ является правильным.

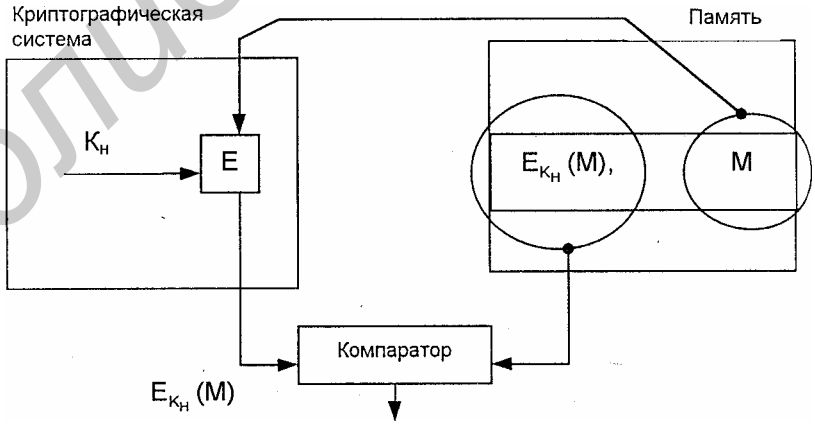

Рис. 3.1. Схема аутентификации мастер-ключа хост-компьютера

Рабочие ключи (например сеансовый) обычно создаются с помощью псевдослучайного генератора и могут храниться в незащищенном месте. Это возможно, поскольку такие ключи генерируются в форме соответствующих

криптограмм, т.е. генератор ПСЧ выдает вместо ключа K_S его криптограмму $E_K(K_S)$, получаемую с помощью мастер-ключа K хост-компьютера. Расшифрование такой криптограммы выполняется только перед использованием ключа K_S.

Схема защиты рабочего (сеансового) ключа показана на рис. 3.2. Чтобы зашифровать сообщение *M* ключом K_S, на соответствующие входы криптографической системы подается криптограмма $E_K(K_S)$ и сообщение *M*. Криптографическая система сначала восстанавливает ключ K_S, а затем шифрует сообщение *M*, используя открытую форму сеансового ключа K_S.

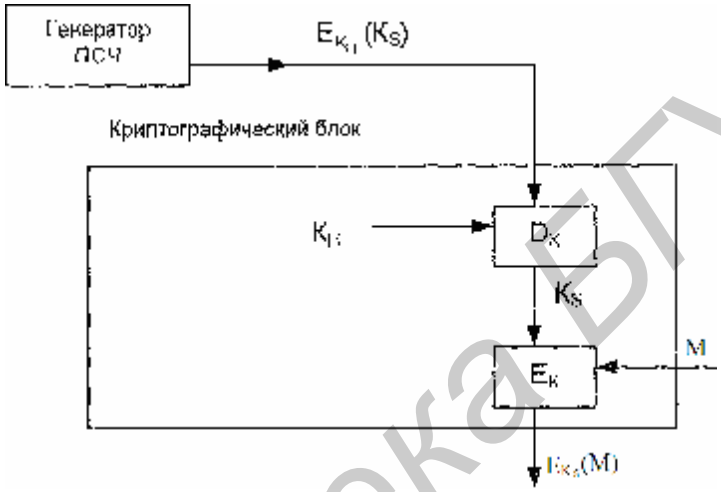

Рис. 3.2. Схема защиты сеансового ключа K_S

Таким образом, безопасность сеансовых ключей зависит от безопасности криптографической системы. Криптографический блок может быть спроектирован как единая СБИС и помещен в физически защищенное место. Очень важным условием безопасности информации является периодическое обновление ключевой информации в криптосистеме. При этом должны переназначаться как рабочие ключи, так и мастер-ключи. В особо ответственных системах обновление ключевой информации (сеансовых ключей) желательно делать ежедневно. Вопрос обновления ключевой информации тесно связан с третьим элементом управления ключами – распределением ключей.

3.3. Распределение ключей

Распределение ключей – самый ответственный процесс в управлении ключами. К нему предъявляются следующие требования:

- оперативность и точность распределения;
- скрытность распределяемых ключей.

Распределение ключей между пользователями компьютерной сети реализуется двумя способами:

- использованием одного или нескольких центров распределения ключей;
- прямым обменом сеансовыми ключами между пользователями сети.

Недостаток первого подхода состоит в том, что центру распределения ключей известно, кому и какие ключи распределены, и это позволяет читать все сообщения, передаваемые по сети. Возможные злоупотребления существенно влияют на защиту. При втором подходе проблема состоит в том, чтобы надежно удостоверить подлинность субъектов сети.

В обоих случаях должна быть обеспечена подлинность сеанса связи. Это можно осуществить, используя уже рассмотренные ранее процедуры аутентификации с использованием механизма запроса–ответа или механизма отметки времени.

Задача распределения ключей сводится к построению протокола распределения ключей, обеспечивающего:

- взаимное подтверждение подлинности участников сеанса;
- подтверждение достоверности сеанса механизмом запроса–ответа или отметки времени;
- использование минимального числа сообщений при обмене ключами;
- возможность исключения злоупотреблений со стороны центра распределения ключей (вплоть до отказа от него).

В основу решения задачи распределения ключей целесообразно положить принцип отделения процедуры подтверждения подлинности партнеров от процедуры собственно распределения ключей. Цель такого подхода состоит в создании метода, при котором после установления подлинности участники сами формируют сеансовый ключ без участия центра распределения ключей с тем, чтобы распределитель ключей не имел возможности выявить содержание сообщений.

3.3.1. Распределение ключей с участием центра распределения ключей

Каждый из участников сеанса А и В имеет мастер-ключ K_i для объекта i ($i = \text{A};\text{B}$), известный только ему и ЦРК. Эти мастер-ключи генерируются в ЦРК и распределяются каждому объекту при личном контакте. Мастер-ключ используется для шифрования сеансового ключа, когда последний передается по сети. Сеансовый ключ K_S генерируется в ЦРК и используется участниками сеанса А и В для защиты сообщений при передаче по линиям связи. Для предотвращения фальсифицированных повторных вызовов на стадии распределения ключей в этом протоколе применяются отметки времени T.

Участник А инициирует фазу распределения ключей, посылая в ЦРК по сети идентификаторы ID_{A} и ID_{B}:

(1) $A \rightarrow$ ЦРК: ID_{A}, ID_{B}, $E_{K_{\text{A}}}(ID_{\text{B}},$ ‘Прошу связь с В’).

Идентификатор ID_{A} посылается в явном виде, поэтому менеджер ЦРК будет знать, какой мастер-ключ необходим для расшифровывания зашифрованной части сообщения. Для этого в ЦРК имеются таблицы идентификаторов и соответствующих им мастер-ключей. Любая попытка злоумышленника изменить ID_{A} приведет к получению неправильного ключа для расшифрования и, следовательно, к выявлению нарушителя службой ЦРК.

Если сообщение правильное, менеджер ЦРК разыскивает мастер-ключ K_{A}, а также вычисляет сеансовый ключ K_S. Затем участнику А посылается ответное сообщение:

(2) ЦРК $\rightarrow A: E_{K_{\text{A}}}\left(T, K_S, ID_{\text{B}}, E_{K_{\text{B}}}\left(T, K_S, ID_{\text{A}}\right)\right)$.

Это сообщение может расшифровать только А, поскольку оно зашифровано ключом K_{A}. Участник А проверяет отметку времени *T*, чтобы убедиться, что это сообщение не является повтором предыдущей процедуры распределения ключей. Идентификатор ID_{A} убеждает А, что действительно требуемый адресат был указан в сообщении (1). Участник А сохраняет у себя сеансовый ключ K_S и посылает участнику В часть сообщения, зашифрованную мастер-ключом объекта В, а также случайное число r_1, зашифрованное сеансовым ключом K_S:

(3) $A \rightarrow B: E_{K_{\text{B}}}\left(T, K_S, ID_{\text{A}}\right), E_{K_S}\left(r_1\right)$.

В этом случае только участник В может расшифровать сообщение (3). Участник В получает отметку времени *T*, сеансовый ключ K_S и идентификатор ID_{A}. Если отметка времени *T* верна, то В уверен, что никто, кроме А, не может быть вызывающим объектом. Фактически верная отметка времени и уникальный идентификатор участника А, зашифрованные ключом K_{B}, обеспечивают подтверждение подлинности А по отношению к В. Далее В извлекает из сообщения случайное число r_1, выполняя расшифрование сеансовым ключом K_S. Для взаимного подтверждения подлинности участник В посылает А сообщение, зашифрованное ключом K_S и содержащее некоторую функцию от случайного числа $f\left(r_1\right)$, например $r_1 - 1$:

(4) $\text{B} \rightarrow \text{A}: E_{K_S}\left(r_1 - 1\right)$.

Если после расшифровывания сообщения (4) участник А получает правильный результат, он знает, что объект на другом конце линии связи действительно В.

Если все шаги успешно выполнены, то этих взаимных подтверждений достаточно, чтобы установить связь, которая будет проходить под сеансовым ключом K_S. Следует отметить, что в этом протоколе необходим обмен с ЦРК

для получения сеансового ключа каждый раз, когда А желает установить связь с В. Данный протокол обеспечивает надежное соединение объектов А и В при условии, что ни один из ключей не скомпрометирован и ЦРК защищен.

Протокол для асимметричных криптосистем с использованием сертификатов открытых ключей

В этом протоколе используется идея сертификатов открытых ключей. Хотя открытые ключи предполагаются известными всем, посредничество ЦРК позволяет подтвердить их подлинность. Без такого посредничества злоумышленник может снабдить А своим открытым ключом, который А будет считать ключом участника В. Затем злоумышленник может подменить собой В и установить связь с А, и его никто не сможет выявить.

Сертификатом открытого ключа *C* называется сообщение ЦРК, удостоверяющее подлинность некоторого открытого ключа объекта. Например, сертификат открытого ключа для пользователя А, обозначаемый C_A, содержит отметку времени Т, идентификатор ID_{A} и открытый ключ K_{A_O}, зашифрованные секретным ключом ЦРК K_C, т.е. $C_{\text{A}} = E_{K_C}\left(T, ID_{\text{A}}, K_{\text{A}_O}\right)$.

Отметка времени Т используется для подтверждения актуальности сертификата и тем самым предотвращает повторы прежних сертификатов, которые содержат открытые ключи и для которых соответствующие секретные ключи несостоятельны.

Секретный ключ ЦРК K_C известен только менеджеру ЦРК. Открытый ключ ЦРК K_O известен участникам А и В. ЦРК поддерживает таблицу открытых ключей всех объектов сети, которые он обслуживает.

Вызывающий объект А инициирует стадию установления ключа, запрашивая у ЦРК сертификат своего открытого ключа и открытого ключа участника В:

(1) А $\rightarrow$ ЦРК: $ID_{\text{A}}, ID_{\text{B}}$, ‘Вышлите сертификаты ключей А и В’.

Здесь – ID_{A}, ID_{B} уникальные идентификаторы соответственно участников А и В.

Менеджер ЦРК отвечает сообщением:

(2) ЦРК $\rightarrow$ А : $E_{K_C}\left(T, ID_{\text{A}}, K_{\text{A}_O}\right), E_{K_C}\left(T, ID_{\text{B}}, K_{\text{B}_O}\right)$.

Участник А, используя открытый ключ ЦРК K_O, расшифровывает ответ ЦРК и проверяет оба сертификата. Идентификатор ID_{B} убеждает А, что личность вызываемого участника правильно зафиксирована в ЦРК и K_{B_O} – действительно открытый ключ участника В, поскольку оба зашифрованы ключом K_C.

Следующий шаг протокола включает установление связи А с В:

(3) А $\rightarrow$ В : $C_{\text{A}}, E_{K_{\text{A}_c}}\left(T\right), E_{K_{\text{B}_o}}\left(r_1\right)$.

Здесь C_{A} – сертификат открытого ключа пользователя А; $E_{K_{\text{A}_c}}(T)$ – отметка времени, зашифрованная секретным ключом участника А и являющаяся подписью участника А, поскольку никто другой не может создать такую подпись; r_1 – случайное число, генерируемое А и используемое для обмена с В в ходе процедуры проверки подлинности. Если сертификат C_{A} и подпись А верны, то участник В уверен, что сообщение пришло от А. Часть сообщения $E_{K_{\text{B}_o}}(r_1)$ может расшифровать только В, поскольку никто другой не знает секретного ключа K_{B_C}, соответствующего открытому ключу K_{B_O}. Участник В расшифровывает значение числа r_1 и, чтобы подтвердить свою подлинность, посылает участнику А сообщение:

(4) $\text{B} \rightarrow \text{A} : E_{K_{\text{A}_o}}(r_1)$.

Участник А восстанавливает значение r_1, расшифровывая это сообщение с использованием своего секретного ключа K_{A_C}. Если это ожидаемое значение r_1, то А получает подтверждение, что вызываемый участник действительно В.

Протокол, основанный на симметричном шифровании, функционирует быстрее, чем протокол, основанный на криптосистемах с открытыми ключами. Однако способность систем с открытыми ключами генерировать цифровые подписи, обеспечивающие различные функции защиты, компенсирует избыточность требуемых вычислений.

3.3.2. Прямой обмен ключами между пользователями

При использовании для информационного обмена криптосистемы с симметричным секретным ключом два пользователя, желающие обменяться криптографически защищенной информацией, должны обладать общим секретным ключом. Для этого пользователи должны обменяться общим ключом по безопасному каналу связи. Если пользователи меняют ключ достаточно часто, то доставка ключа превращается в серьезную проблему.

Для решения этой проблемы можно применить два способа:

1) использование криптосистемы с открытым ключом для шифрования и передачи секретного ключа симметричной криптосистемы;

2) использование системы открытого распределения ключей Диффи–Хеллмана.

Первый способ был уже рассмотрен ранее. Второй способ основан на применении системы открытого распределения ключей. Эта система позволяет пользователям обмениваться ключами по незащищенным каналам связи.

Алгоритм открытого распределения ключей Диффи–Хеллмана

Алгоритм Диффи–Хеллмана был первым алгоритмом с открытыми ключами (предложен в 1976 г.). Его безопасность обусловлена трудностью вычисления дискретных логарифмов в конечном поле, в отличие от легкости дискретного возведения в степень в том же конечном поле.

Предположим, что два пользователя А и В хотят организовать защищенный коммуникационный канал.

1. Обе стороны заранее уславливаются о модуле n (n должно быть простым числом, $(n-1)/2$ также должно быть простым) и элементе g, $(1 \le g \le n-1)$. Эти числа могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей системы.

2. Затем пользователи А и В независимо друг от друга выбирают собственные секретные ключи X_{A} и X_{B} (X_{A}, X_{B} – случайные большие целые числа, которые хранятся пользователями А и В в секрете).

3. Далее пользователь А вычисляет открытый ключ $Y_{\text{A}} = g^{X_{\text{A}}} (\text{mod}\, n)$, а пользователь В – открытый ключ $Y_{\text{B}} = g^{X_{\text{B}}} (\text{mod}\, n)$.

4. Затем стороны А и В обмениваются вычисленными значениями открытых ключей Y_{A} и Y_{B} по незащищенному каналу. (Считаем, что все данные, передаваемые по незащищенному каналу связи, могут быть перехвачены злоумышленником.)

5. Далее пользователи А и В вычисляют общий секретный ключ: пользователь А: $K = (Y_{\text{B}})^{X_{\text{A}}} = (g^{X_{\text{B}}})^{X_{\text{A}}} \bmod n$, пользователь В: $K' = (Y_{\text{A}})^{X_{\text{B}}} = (g^{X_{\text{A}}})^{X_{\text{B}}} \bmod n$. При этом $K = K'$, так как $(g^{X_{\text{B}}})^{X_{\text{A}}} \bmod n = (g^{X_{\text{A}}})^{X_{\text{B}}} \bmod n$.

Ключ К может использоваться в качестве общего секретного ключа (ключа шифрования ключей) в симметричной криптосистеме.

Алгоритм открытого распределения ключей Диффи–Хеллмана позволяет обойтись без защищенного канала для передачи ключей. Однако, работая с этим алгоритмом, необходимо иметь гарантию того, что пользователь А получил открытый ключ именно от пользователя В, и наоборот. Эта проблема решается с помощью электронной подписи, которой подписываются сообщения об открытом ключе.

Метод Диффи-Хеллмана дает возможность шифровать данные при каждом сеансе связи на новых ключах. Это позволяет не хранить секреты на дискетах или других носителях. Не следует забывать, что любое хранение секретов повышает вероятность попадания их в руки конкурентов или противника.

ЛИТЕРАТУРА

1. Закон РБ от 6 сентября 1995 г. «Об информации».
2. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования.
3. СТБ 1176.2-99 Информационная технология. Защита информации. Процедуры выработки и проверки электронной цифровой подписи.
4. СТБ 1176.1-99 Информационная технология. Защита информации. Функция хэширования.
5. Романец, Ю. В. Защита информации в компьютерных системах и сетях / Ю. В. Романец, П. А. Тимофеев, В. Ф. Шаньгин. – М. : Радио и связь, 1999.
6. Основы криптологии / Ю. С. Харин [и др.]. – Минск : Новое знание, 2003.
7. Шнайдер, Б. Прикладная криптография / Б. Шнайдер. – М, 2001.
8. Зима, В. М. Безопасность глобальных сетевых технологий / В. М. Зима, А. А. Молдовян, Н. А. Молдовян. – СПб, 2001.
9. Мафтик, С. Механизмы защиты в сетях ЭВМ / С. Мафтик. – М. : Мир, 1993.
10. Правовые и организационно-технические методы защиты информации: учеб.пособие / В. Ф. Голиков [и др.]. – Минск : БГУИР, 2004.
11. Голиков, В. Ф. Криптографическое кодирование информации : метод. указания к лабораторным работам / В. Ф. Голиков, А. В. Курилович. – Минск : БГУИР, 2002.

Св. план 2008, поз. 90

Учебное издание

Голиков Владимир Федорович
Курилович Андрей Владимирович

КРИПТОГРАФИЧЕСКАЯ ЗАЩИТА ИНФОРМАЦИИ В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

Учебно-методическое пособие

для студентов специальностей «Сети телекоммуникаций»
и «Защита информации в телекоммуникациях»
всех форм обучения

В 2-х частях
Часть 2

Редактор М. В. Тезина
Корректор Е. Н. Батурчик

Подписано в печать 21.04.2008. Формат 60×84 1/16. Бумага офсетная.
Гарнитура «Таймс». Печать ризографическая. Усл. печ. л. 1,98.
Уч.-изд. л. 1,6. Тираж 100 экз. Заказ 17.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0056964 от 01.04.2004. ЛП №02330/0131666 от 30.04.2004.
220013, Минск, П. Бровки, 6