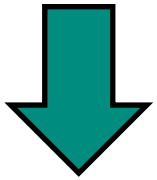
ЭЛЕКТРОННЫЕ ДЕНЬГИ

Возможности криптографии используются в реальных компьютерных системах:

- оплата покупок при помощи электронных карточек
- заказ авиабилетов и билетов через Интернет
- покупки товаров в Интернет-магазинах
- многое другое

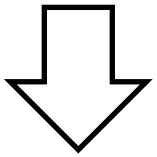
Сведения о покупках накапливаются в магазинах и банках



«проблема Большого Брата»

«Проблема Большого Брата»:

Исчезает анонимность процесса покупки - информация о покупках любого человека может стать известной третьим лицам и использоваться против него.



Возникла идея разработать такие схемы электронных платежей, которые сохраняли бы анонимность покупателя в той же степени, что и при расчете наличными деньгами.

Такие протоколы называются **электронными или цифровыми деньгами (digital cache).**

- В 1990-х годах шифрование на уровне протокола только начинало развиваться, люди боялись передавать данные своих карт по незащищенному каналу.

- Возник большой интерес к промежуточной архитектуре и появились компании, которые стали посредниками в платежах.

FirstVirtual - первая промежуточная платежная компания. Затем появились **CyberCash** и другие.

FirstVirtual, CyberCash и другие

В подобных платежных системах общение между покупателем и продавцом происходило через компанию-посредника.

Покупатель отправлял информацию о своей кредитной карте этому посреднику, который утверждал транзакцию и уведомлял продавца.

Посредник рассчитывался с продавцом в конце каждого дня.

FirstVirtual, CyberCash и другие

Достоинства:

- Гарантированная анонимность покупателя перед продавцом.
- Данные кредитных карт покупателей остаются в безопасности.

Недостаток:

- Потеря удобства непосредственного общения покупателя с продавцом

Электронные (или цифровые) деньги - Digital cache

1983 г.

Дэвид Чаум

(David Chaum)

изобрел алгоритм «слепой» электронной подписи и впервые реализовал его на основе метода RSA

Электронные деньги

Протокол имеет 2 дополнительных преимущества:

1. **Анонимность**, поскольку людям не нужно использовать свою настоящую личность для оплаты покупок, и банки не могут отслеживать все расходы пользователей, как они это делали по кредитным картам.
2. **Электронные деньги позволяют проводить офлайн-транзакции**, когда нет необходимости вызывать третье лицо для подтверждения транзакции.

Рассмотрим **две «плохие» схемы**, а затем **«хорошую»**, исключающую недостатки и уязвимости «плохих» схем.

В данной схеме три участника: **продавец**, **покупатель** и **банк**. У продавца и покупателя имеются счета в данном банке.

Чтобы осуществить куплю-продажу товара или услуги, нужно провести процесс, состоящий из трех этапов:

1. Снятие покупателем необходимой суммы со своего счета в банке;
2. «Пересылка» этой суммы от покупателя к продавцу;
3. Уведомление банка (о совершении сделки) продавцом, зачисление банком нужной суммы на счет магазина и доставка товара покупателю.

Наша цель — разработать такую схему, чтобы

- она была надежна;
- чтобы банк не знал, кто купил товар, т.е. была сохранена анонимность обычных денег.

Опишем первую «плохую» схему (она базируется на RSA). Банк имеет следующую информацию: секретные числа P, Q, c и открытые

$$N = PQ,$$

$$d = c^{-1} \bmod (P-1)(Q-1). \tag{5.9}$$

Допустим, покупатель решил израсходовать некоторую заранее оговоренную с банком сумму (например, 100$). (Сначала рассмотрим случай, когда может использоваться банкнота только одного номинала).

Покупатель высылает в банк число n , которое будет номером банкноты (обычно требуется, чтобы генерировалось случайное число в промежутке [2, N - 1]).

Банк вычисляет число

$$s = n^c \bmod N \qquad (5.10)$$

и формирует банкноту $< n, s >$, которую возвращает покупателю, предварительно уменьшив его счет на 100$.

Параметр ***s*** в банкноте — это подпись банка. Никто не может подделать подпись, так как число ***c*** - секретно.

Покупатель предъявляет банкноту $< n, s >$ в магазине, чтобы купить товар. Магазин отправляет эту банкноту в банк для проверки.

Банк проверяет правильность подписи (эту проверку мог бы сделать и магазин, используя открытые ключи банка).

Но кроме этого банк хранит все номера возвратившихся к нему банкнот и проверяет, нет ли числа ***n*** в этом списке.

Если ***n*** есть в списке, то платеж не принимается (кто-то пытается использовать банкноту повторно), и банк сообщает об этом магазину.

Если же все проверки прошли успешно, то банк добавляет 100$ на счет магазина, а магазин отпускает товар покупателю.

Недостаток этой схемы — отсутствует анонимность. Банк, а также все, кто имеет доступ к открытым линиям связи, могут запомнить, какому покупателю соответствует число ***n*** , и тем самым выяснить, кто купил товар.

Рассмотрим вторую «плохую» схему, которая уже обеспечивает анонимность. Эта схема базируется на так называемой «слепой подписи».

Снова покупатель хочет купить товар. Он генерирует число n, которое теперь *не будет* посылаться в банк. Затем он генерирует случайное число r, взаимно простое с N, и вычисляет число

$$\hat{n} = (n \cdot r^d) \bmod N. \tag{5.11}$$

Число $\hat{n}$ покупатель отправляет в банк.

Банк вычисляет число

$$\hat{s} = \hat{n}^c \bmod N \tag{5.12}$$

и отправляет $\hat{s}$ обратно покупателю (не забыв при этом снять 100\$ с его счета).

Покупатель находит число $r^{-1} \bmod N$ и вычисляет

$$s = (\hat{s} \cdot r^{-1}) \bmod N. \tag{5.13}$$

Заметим, что с учетом соотношений (5.12), (5.11) и (5.9) имеем

$$s = \hat{n}^c \cdot r^{-1} = (n \cdot r^d)^c \cdot r^{-1} = n^c r^{dc} \cdot r^{-1} = n^c r^1 r^{-1} = n^c \bmod N,$$

т.е. мы получили подпись банка к n (см. (5.10)), но самого числа n ни банк, ни кто либо другой не видел. Вычисление (5.12) называется «слепой подписью», так как реальное сообщение (n) подписывающий не видит и узнать не может.

Таким образом, покупатель имеет число n, которое никому не известно и никогда не передавалось по каналам связи, и подпись банка s, совпадающую с вычисленной по (5.10). Покупатель формирует банкноту $\langle n,\ s\rangle$ и действует так же, как в первой «плохой» схеме.

Но теперь никто не знает, кому соответствует эта банкнота, т.е. она стала анонимной, как обычная бумажная банкнота.

Действия магазина и банка после предъявления покупателем банкноты $\langle n,\ s\rangle$ ничем не отличаются от действий, описанных в первой схеме.

Почему же данная схема плохая? Она имеет следующий недостаток: можно сфабриковать фальшивую банкноту, если известны хотя бы две настоящие. Делается это так. Путь злоумышленник (будь то покупатель или магазин) имеет две настоящие банкноты $\langle n_1,\ s_1\rangle$ и $\langle n_2,\ s_2\rangle$. Тогда он легко сможет изготовить фальшивую банкноту $\langle n_3,\ s_3\rangle$, вычислив числа

$$n_3 = n_1 n_2 \bmod N,$$
$$s_3 = s_1 s_2 \bmod N.$$

Действительно,

$$n_3^c = (n_1 n_2)^c = n_1^c n_2^c = s_1 s_2 = s_3 \bmod N, \tag{5.14}$$

т.е. s_3 является правильной подписью для n_3, и у банка нет никаких оснований, чтобы не принять эту фальшивую банкноту (он просто не сможет отличить ее от подлинной). Это так называемое «мультипликативное свойство» системы RSA.

Опишем, наконец, «хорошую» схему, в которой устранены все недостатки первых двух. В одном варианте такой схемы используется некоторая односторонняя функция

$$f : \{1, \ldots, N\} \to \{1, \ldots, N\}$$

(f вычисляется легко, а обратная к ней функция f^{-1} — очень трудно). Функция f не секретна и известна всем (покупателю, банку и магазину).

Банкнота теперь определяется как пара чисел $\langle n,\ s_f \rangle$, где

$$s_f = (f(n))^c \bmod N,$$

т.е. подписывается не n, а значение $f(n)$.

При выборе односторонней функции нужно проявлять осторожность.

Например, функция $f(n) = n^2 \bmod N$, которая действительно является односторонней, не годится для рассматриваемого протокола.

Банкноты, созданные с использованием такой функции, будут по-прежнему обладать мультипликативным свойством (5.14). На практике в качестве $f(n)$ обычно используются криптографичесие хеш-функции.

Все остальные действия магазина и банка остаются такими же, как и в ранее описанных схемах.

Более простой способ борьбы с мультипликативным свойством системы RSA — **внесение избыточности в сообщение.**

Допустим, что длина модуля ***N*** — 1024 бита. Такой же может быть и длина числа ***n***.

Будем записывать (случайно выбираемый) номер банкноты только в младшие 512 бит ***n***, а в старшие 512 бит ***n*** запишем некоторое фиксированное число.

Это фиксированное число может нести полезную информацию, такую, как номинал банкноты и наименование банка (с помощью 512 бит можно представить строку из 64 символов ASCII).

Теперь банк при предъявлении ему банкноты будет обязательно проверять наличие фиксированного заголовка в параметре ***n*** и отвергать банкноту в случае его отсутствия.

Вероятность того, что при перемножении двух чисел по модулю ***N*** результат совпадет с ними в 512 битах пренебрежимо мала. Поэтому получить фальшивую банкноту по формуле (5.14) не удастся.

Пример 5.4. Пусть в качестве секретных параметров банка выбраны числа $P = 17$, $Q = 7$, $c = 77$. Соответствующие им открытые параметры будут $N = 119$, $d = 5$. Для исключения возможности подделки банкнот их допустимыми номерами считаются только числа, состоящие из двух одинаковых десятичных цифр, например, 11, 77, 99.

Когда покупатель хочет получить банкноту, он вначале случайным образом выбирает ее номер (из числа допустимых). Предположим, он выбрал $n = 33$. Затем он находит случайное число r, взаимнопростое со 119. Допустим, $r = 67$, $\gcd(67, 119) = 1$. Далее, покупатель вычисляет

$$\hat{n} = (33 \cdot 67^5) \bmod 119 = (33 \cdot 16) \bmod 119 = 52.$$

Именно число 52 он посылает в банк.

Банк списывает со счета покупателя 100\$ и отправляет ему число

$$\hat{s} = 52^{77} \bmod 119 = 103.$$

Покупатель вычисляет $r^{-1} = 67^{-1} \bmod 119 = 16$ и $s = 103 \cdot 16 \bmod 119 = 101$ и получает платежеспособную банкноту

$$\langle n, s \rangle = \langle 33, 101 \rangle.$$

Эту банкноту он приносит (или посылает) в магазин, чтобы купить товар.

Магазин предъявляет банкноту в банк. Банк делает следующие проверки:

1) номер банкноты ($n = 33$) состоит из двух одинаковых десятичных цифр (т.е. содержит требуемую избыточность);

2) ранее банкнота с таким номером не предъявлялась;

2) подпись банка верна, т.е. $33^5 \bmod 119 = 101$.

Так как все проверки прошли успешно, банк зачисляет 100\$ (это фиксированный номинал банкноты) на счет магазина, о чем ему и сообщает. Магазин отпускает товар покупателю.

Еще две проблемы, возникающие в связи с рассмотренной схемой электронных денег.

1. Независимо действующие покупатели или даже один покупатель, который не помнит номеров ранее использованных им банкнот, могут случайно сгенерировать две или более банкноты с одинаковыми номерами.

По условиям протокола банк примет к оплате только одну из таких банкнот (ту, которая будет предъявлена первой). Однако если номер банкноты — число длиной 512 бит и покупатели генерируют его действительно случайно, то вероятность получения когда либо двух одинаковых номеров мала.

2. В рассмотренной схеме используются только банкноты одного фиксированного номинала, что, конечно, неудобно для покупателя.

Решение проблемы использования банкнот разного номинала возможно следующим образом.

Банк заводит несколько пар (c_i, d_i), обладающих свойством (5.9), и объявляет, что d_1 соответствует, например, 1000 руб., d_2 — 500 руб. и т.д.

Когда покупатель запрашивает слепую подпись в банке, он дополнительно сообщает, какого номинала банкноту он хочет получить.

Банк снимает с его счета сумму, равную указанному номиналу, и формирует подпись, используя соответствующее секретное число c_i.

Когда впоследствии банк получает подписанную банкноту, он использует для проверки подписи по очереди числа d_1, d_2 и т.д.

Если подпись оказалась верна для какого-то d_i, то принимается банкнота i-го номинала. В случае, когда параметр n банкноты содержит фиксированный заголовок с указанием ее номинала, задача проверки подписи облегчается — банк сразу использует нужный ключ d_i.