Асимметричная криптография

Авторы – У. Диффи, М.Хеллман – 1976 г.

Идея – использовать ключи парами (**К1**: для заш и **К2**: для расш), которые очень трудно вычислить один из другого; ключ **К1** известен и доступен для всех, ключ **К2** - тайный

Упрощенная схема преобразования (Аня – **А** - передает зашифрованное сообщение Васе - **В**):

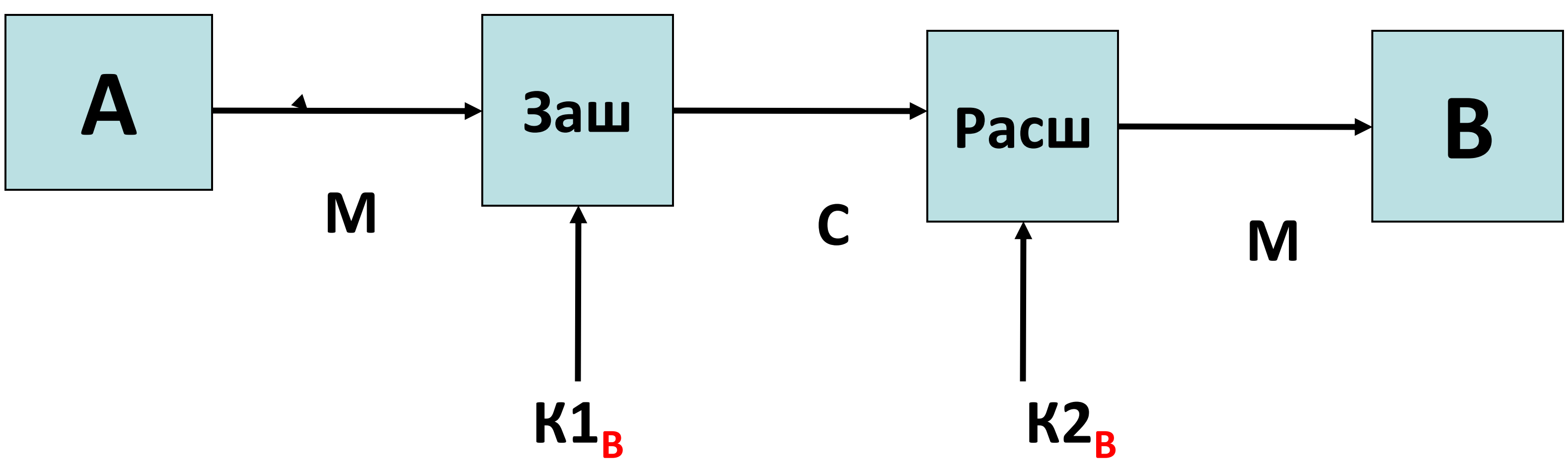


Рис.1

Асимметричная криптография

Разложение больших чисел на множители (**p** и **q**)

Число бит	**MIPS-лет**
512	30 тыс

Рекомендации Ривеста по выбору длины ключа (мин – макс)

Год	Длина, бит
1	398 -1289
2	405 – 1399
3	422 – 1512
4	439 – 1628
5	455 - 1754
2020	489 - 2017

Асимметричная криптография

Длины симметр и асимметр ключей с равной устойчивостью к лобовому (путем перебора) вскрытию

Сим ключ, бит	Асим ключ, бит
56	384
64	512
80	768
112	1792
128	2304

Алгоритм обмена ключами Диффи-Хеллмана

Алгоритм Диффи-Хеллмана (Whitfield Diffie, Martin Hellman): генерация секретного ключа двумя сторонами (1976)

Основная идея: **использовать ключи взаимосвязанными парами: один – для зашифрования, другой – для расшифрования, которые невозможно вычислить один на основе другого**

A и **B** выбирают совместно большие простые ч: **n** и **g**

Прот-л обмена:

1.**A** выб-т случ число **x**, выч-т $X = g^x \bmod n$ и отсылает **X** **B**

2.**B** выб-т случ число **y**, выч-т $У = g^y \bmod n$ и отсылает **У** **A**

3. **A** выч-т $k1 = Y^x \bmod n$

4. **B** выч-т $k2 = X^y \bmod n$

Получается: $k1 = k2 = g^{xy} \bmod n = k$ **–** *секретный ключ*

Зная **n, g, X, Y**, невозможно при больших значениях чисел определить

k – проблема дискр. логарифма

Проблема: **атака человек-в-середине**

- **Как работает алгоритм, если число согласующих сторон >2?**

Основной недостаток алгоритма Д-Х

Протокол Диффи-Хеллмана является уязвимым для атаки, называемой **"человек в середине".**

Злоумышленник **С** может перехватить открытое значение, посылаемое от **A** к **B**, и послать вместо него свое открытое значение.

Затем он может перехватить открытое значение, посылаемое

от **B** к **A**, и также послать вместо него свое открытое значение.

Тем самым **С** получит общие секретные ключи с **А** и **В** и сможет читать и/или модифицировать сообщения, передаваемые от одной стороны к другой.

Функция Эйлера

Если – **n -** простое число, то **ф(n) = n-1.**

Если **n=p*q**, где **p** и **q** – простые числа, то

ф(n) = (p-1)*(q-1)

Малая теорема Ферма. Если **n** – простое число и **a** не кратно **n**, то справедливо

$$a^{(n-1)} \equiv 1 \bmod n.$$

Обобщение Эйлера МТФ: Если НОД (**a**, **n**) =1, то

$$a^{\phi(n)} \bmod n = 1$$

Нетрудно вычислить a^{-1}:

$$a^{-1} = a^{\phi(n)-1} \bmod n$$

Пример 4. Найти число, обратное 5 по модулю 7.

Ответ: ф(n)-1 =7-1=6; 5^{6-1} mod 7= 3.

Алгоритм на основе задачи об укладке ранца

Задача о рюкзаке или ранце (*knapsack*) является классической задачей дискретной оптимизации.
Задача относится к числу **NP**-полных.

Данная задача и ее варианты широко используются для моделирования большого числа практических задач.

В общем виде задачу можно сформулировать так:
из заданного множества предметов m_i *общим числом* z *со свойствами «стоимость» и «вес»* S*, требуется отобрать некоторое число предметов таким образом, чтобы получить максимальную суммарную стоимость при одновременном соблюдении ограничения на суммарный вес.*

Один и тот же предмет не может быть взят несколько раз.

	1	2	3	4	5	6	7	8
Вес	1	5	6	11	14	20	32	43
Стои-мость	18	20	17	19	25	21	27	23

$z = 10$

Более строго задача формулируется так:

дан набор значений $\mathbf{m_1}$, $\mathbf{m_2}$, **...**, $\mathbf{m_n}$ и суммарное значение $\mathbf{S}$; требуется вычислить значения $\mathbf{b_i}$ такие что

$$\mathbf{S = b_1 m_1 + b_2 m_2 + \ldots + b_z m_z},$$

где $\mathbf{z}$ – количество предметов; $\mathbf{b_i}$ - бинарный множитель. Значение $b_i = 1$ означает, что предмет i кладут в рюкзак, $b_i = 0$ - не кладут.

Например, веса предметов имеют значения **1, 5, 6, 11, 14, 20, 32 и 4**3. При этом можно упаковать рюкзак так, чтобы его вес **S** стал равен **22**, использовав предметы весом **5, 6 и 11**.

Невозможно упаковать рюкзак так, чтобы его вес стал равен 24.

Р. Меркл и М. Хеллман предложили использовать ранцевый алгоритм в качестве алгоритма шифрования с открытым ключом.

- В основе алгоритма Меркла-Хеллмана лежит идея : **предметы k_i из кучи общим числом z выбираются с помощью блока открытого текста, длина которого (в битах) равна количеству z предметов в куче.**
- **При этом биты открытого текста соответствуют значениям b_i, а текст является полученным суммарным весом S.**

Пример шифрограммы, полученной с помощью задачи об укладке ранца, показан в следующей таблице.

Открытый текст	1 1 1 0 0 1 0 0	0 1 0 1 1 0 0 1	0 0 0 0 0 0 0 0
Рюкзак (ключ)	1 5 6 11 14 20 32 43	1 5 6 11 14 20 32 43	1 5 6 11 14 20 32 43
Шифрограмма	32 (1+5+6+20)	73 (5+11+14+43)	0

Открытый текст состоит из трех байтов.
Шифрограмма: 32, 73, 0

Суть метода для шифрования состоит в том, что существуют две различные задачи укладки ранца - одна из них решается легко и характеризуется линейным ростом трудоемкости, а другая - нет.

Легкий для укладки ранец можно превратить в трудный:
можно применить в качестве **открытого ключа трудный** для укладки ранец, который легко использовать для зашифрования, но невозможно - для дешифрования.
в качестве **закрытого ключа** применить **легкий** для укладки ранец, который предоставляет простой способ расшифрования сообщения.

В качестве закрытого ключа (легкого для укладки ранца) используется **сверхвозрастающая последовательность**.

Сверхвозрастающей называется **последовательность**, в которой каждый последующий член больше суммы всех предыдущих.

Пример. Последовательность {2, 3, 6, 13, 27, 52, 105, 210} является сверхвозрастающей, а {1, 3, 4, 9, 15, 25, 48, 76} - нет.

Решение для **сверхвозрастающего ранца** найти легко:

в качестве текущего выбирается полный вес ***S***, который надо получить, и сравнивается с весом самого тяжелого предмета в ранце (k_z);

если текущий вес меньше веса данного предмета, то его в рюкзак не кладут, в противном случае его укладывают в рюкзак;

уменьшают текущий вес на вес положенного предмета и переходят к следующему по весу предмету в последовательности.

Шаги повторяются до тех пор, пока процесс не закончится.

Если текущий вес уменьшится до нуля, то решение найдено. В противном случае, нет.

Пример. Пусть полный вес рюкзака равен 270 (**S**=270), а последовательность весов предметов равна
{2, 3, 6, 13, 27, 52, 105, 210} (**k_1** =2, **k_2** =3, **k_3** =6 и т.д.).

- Самый большой вес – 210. Он меньше 270, поэтому предмет весом 210 кладут в рюкзак (1).
- Вычитают 210 из 270 и получают 60.
- Следующий наибольший вес последовательности равен 105. Он больше 60, поэтому предмет весом 105 в рюкзак не кладут (0).

Следующий самый тяжелый предмет имеет вес 52. Он меньше 60, поэтому предмет весом 52 также кладут в рюкзак (1).

- Аналогично проходят процедуру укладки в рюкзак предметы весом 6 и 2.

В результате полный вес уменьшится до 0.
Если бы этот рюкзак был использован для расшифрования, то открытый текст, полученный из значения шифртекста 270, был бы равен 10100101.

Открытый ключ представляет собой **нормальную** (не сверхвозрастающую) последовательность.

Он **формируется на основе закрытого ключа** и не позволяет легко решить задачу об укладке ранца.

Для его получения все значения закрытого ключа умножаются на число **a** по модулю **n**. Значение модуля **n** должно быть больше суммы всех чисел последовательности, например, **420** (2+3+6+13+27+52+105+210=418).

Множитель **a** должен быть взаимно простым числом с модулем **n**, например, a=**31**.

Результат построения нормальной последовательности (открытого ключа) представлен в следующей таблице.

Закрытый ключ, k_i	2	3	6	13	27	52	105	210
Открытый ключ	62	93	186	403	417	352	315	210

Открытый ключ: k_i *a (mod **n) = k_i * 31 (mod 420)**

Зашифрование сообщения на основе задачи об укладке ранца

- Для зашифрования сообщения оно сначала разбивается на блоки, по размерам равные числу (**z**) элементов последовательности в рюкзаке.
- Затем, считая, что **1** указывает на присутствие элемента последовательности в рюкзаке, а **0** — на его отсутствие, вычисляются полные веса рюкзаков: **по одному рюкзаку для каждого блока сообщения с использованием открытого ключа Получателя)**.

Пример. Возьмем открытое сообщение **М**, состоящее из 7 букв (m_i), которые представим в бинарном виде (1 символ текста – 1 байт). Бинарное представление символов дано в первом столбце нижеследующей таблицы.

Открытый ключ: {62, 93, 186, 403, 417, 352, 315, 210}

Результат зашифрования каждого бока (буквы) сообщения с помощью открытого ключа представлен в правом столбце .

m_i		c_i
1100 0000	62+93	155
1100 0001	62+93+210	365
1101 0000	62+93+403	558
1100 0000	62+93	155
1100 1100	62+93+417+352	924
1100 1110	62+93+417+352+315	1239
1100 0010	62+93+315	470

Шифртекст: 155 365 558 155 924 1239 470

Расшифрование сообщения на основе задачи об укладке ранца

- Для расшифрования сообщения **получатель** (использует свой тайный ключ: сверхвозрастающую последовательность) должен сначала определить обратное число **a^{-1}**, такое что

$$(a * a^{-1}) \bmod n = 1.$$

Для вычисления обратных чисел по модулю можно использовать расширенный алгоритм Евклида.

- После определения обратного числа каждое значение шифрограммы умножается на $a^{-1} \bmod n$ и с помощью закрытого ключа определяются биты открытого текста.

В нашем **примере** значение **a^{-1}** = 271: (31*271 mod 420 = 1).

В примере сверхвозрастающая последовательность равна {2, 3, 6, 13, 27, 52, 105, 210}, **n** = 420, **a** = 31.

Шифртекст: 155 365 558 155 924 1239 470

Расшифрование первого блока шифртекста c_1=155:

сначала умножаем $c_1 * a^{-1} \pmod{n} = 155*271 \pmod{420} = 5$

Используем **5**, как параметр **S**, и с помощью сверх-возрастающей последовательности ({2, 3, 6, 13, 27, 52, 105, 210}) получаем **m_1= 11000000**

Алгоритм **RSA**

Первый полноценный алгоритм с открытым ключом.

Авторы: Рон Райвест (Ron Rivest), Ади Шамир (Adi Shamir), Леонард Эдельман (Leonard Adelman)

Ключ – большие простые числа: **e, d, n**

Генерация ключа**:** два больших целых числа: **p, q** (см. лекцию: ф. Эйлера): **p* q = n**

Случайно выбирается **e** (**e** и **φ(n)**=**(p-1)* (q -1)** – взаимно простые числа)

С пом. ф-и Эйлера находим **d**: $\mathbf{e*d=1 \bmod \varphi(n)}$ (1)

или $\mathbf{d=e^{-1} \bmod \varphi(n);}$ (2)

e и **d –** также взаимно простые числа

e и **n** – открытый (публичный) ключ (**К1**), **d** (**d** и **n**) **-** закрытый (тайный) ключ (**К2**)

Защифрование: $\mathbf{c_i = (m_i)^{e}* \bmod n}$ (3)

Расшифрование: $\mathbf{m_i = (c_i)^{d} \bmod n}$ (4)

<u>Пример.</u> Пусть **p=47** и **q=71,** тогда **n = 3337**

Ключ **e** не должен иметь общих множителей с **(p-1)* (q -1) = 46*70=3220;** принимаем **e=79,** тогда

$\mathbf{d= 79^{-1} \bmod 3220 = 1019}$

Пусть **M=688232687** и длина блока равна **3:**

$\mathbf{m^1=688, m^2=232, m^3=687}$

Зашифрование (по (3)):$\mathbf{c^1= 688^{79} \bmod 3337=1570}$ и т.д.

Расшифрование (по (4)):$\mathbf{m^1=1570^{1019} \bmod 3337=688}$ и т.д.

Стойкость RSA

- Стойкость RSA основывается на большой вычислительной сложности известных алгоритмов разложения произведения простых чисел на сомножители.

Пример. Легко найти произведение двух простых чисел 7 и 13 – 91. Попробуйте два простых числа, произведение которых равно 91 (7 и 13) или 323 (числа 17 и 19).

Для надежного шифрования алгоритмом RSA, как правило, выбираются простые числа, количество двоичных разрядов которых составляет тысячи.

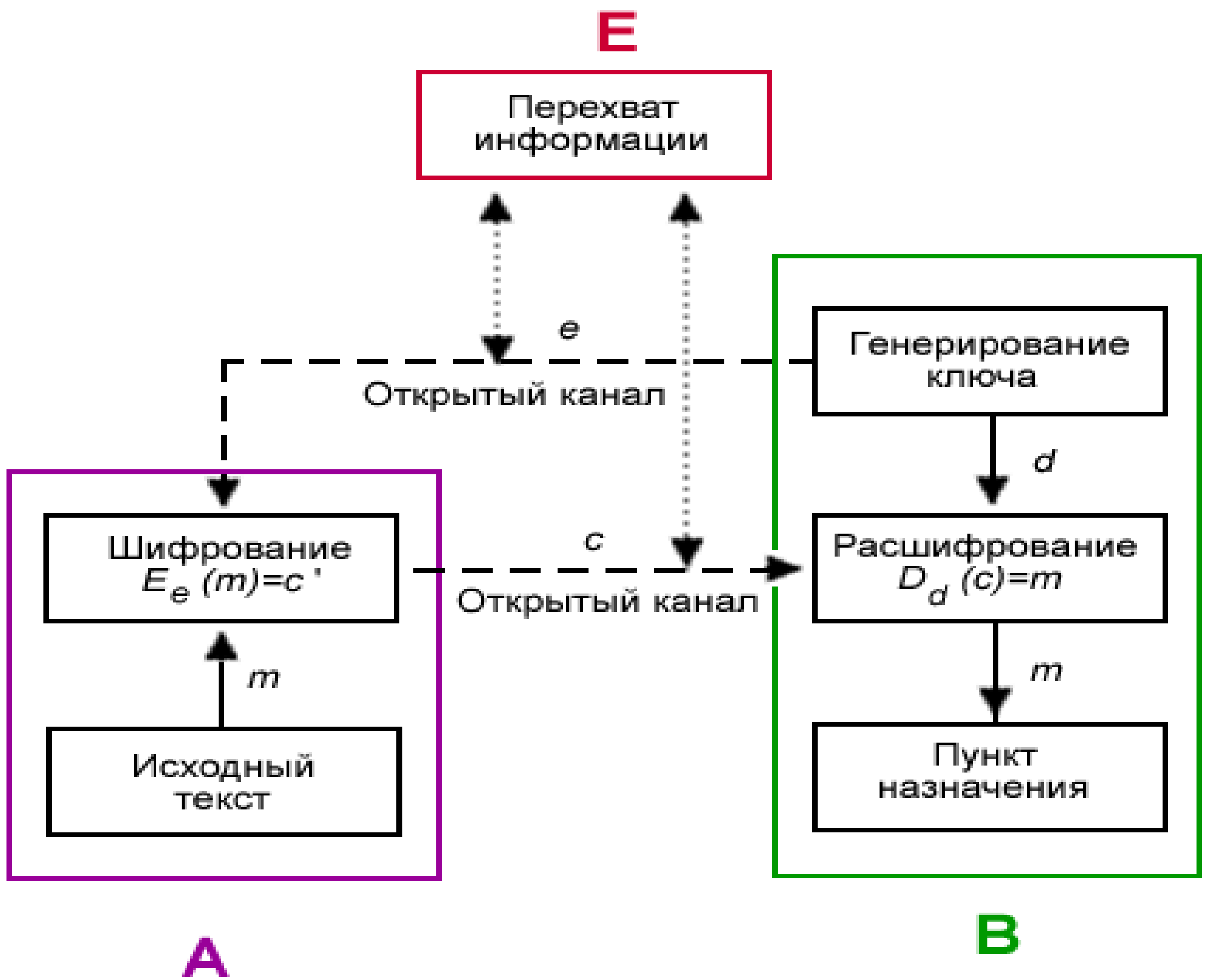
Атаки на **RSA.**

Криптоаналитик имеет шифртекс $\mathbf{C= E_e(M).}$

Цель – восстановить **M**(расшифровать **C**).

Метод: Шифрует известным открытым ключом произвольное сообщение **M'**. Получает **C'**. Если **C=C',** то получено **M (M=M')**, в ином случае шифруется другое сообщение **M''** и т.д.

Общая схема системы криптоанализа с перехватом



- Такую возможность исключает вероятностное шифрование (алгоритмы Эль-Гамаля, алгоритмы на основе эллиптических кривых – <u>зашифрование одного и того же сообщения одним и тем же открытым ключом дает различные шифртексты. Но при расшифровании – получаются одинаковые сообщения</u>.

Асимметричный алгоритм шифрования Эль-Гамаля

- Стойкость алгоритма базируется **на сложности решения задачи дискретного логарифмирования**.
- Предложен в 1985г.

1. Генерация ключа

№ п/п	Описание операции	Пример
1	Выбираются простое число **p**.	p=23
2	Выбираются произвольное число **g**, являющееся первообразным корнем по модулю **p**. **Первообразный корень по модулю p** – наименьшее положительное целое число **g** такое, что $g^{\varphi(p)} \bmod p = 1$ и $g^{i} \bmod p \neq 1, \quad \text{для } 1 \leq i < \varphi(p)$ где $\varphi(p)$ – функция Эйлера. Т.к. **p** – простое число, то $\varphi(p) = p - 1$.	g=5
3	Выбираются случайное целое число **x** ($1 < x < p$).	x=3
4	Вычисляется **y** = $g^x \bmod p$	$y = 5^3 \bmod 23 = 125 \bmod 23 = 10$
5	Открытый ключ - **y, g** и **p**. Причем g и p можно сделать общими для группы пользователей. Закрытый ключ - **x**.	

Зашифрование (каждого отдельного блока исходного сообщения)

а) **выбирается случайное число k (1 < k < p – 1)**,

б) шифрограмма (**C**) генерируется по следующим формулам:

$$\mathbf{a = g^k \bmod p,} \qquad (5)$$

$$\mathbf{b = (y^k * M) \bmod p,} \qquad (6)$$

где **M** – исходное сообщение;

(a, b) – зашифрованное сообщение (**C**).

Расшифрование **C** выполняется по следующей формуле:

$$\mathbf{M = (b * (a^x)^{-1}) \bmod p} \qquad (7)$$

или

$$\mathbf{M = (b * a^{p-1-x}) \bmod p,} \qquad (8)$$

где $(\mathbf{a}^x)^{-1}$ – обратное значение числа $\mathbf{a}^x$ по модулю **p**.

Пример. M= «Абрамов» (01 02 18 01 14 16 03)

Ключ: открытый: y=10, g=5, p=23

тайный: x=3

Зашифрование. Выбираем **k=7** (для шифрования каждого символа следует выбирать разные **k** – принцип вероятностного шифра).

Для упрощения примем **k** неизменным.

Используем (5): $\mathbf{a = g^k \bmod p =} 5^7 \bmod 23 = 17$ (для каждого символа д.б. подсчитано отдельное знач. **a**

Используем (6): $\mathbf{b = (y^k * M) \bmod p}$ для вычисления второй части шифртекста:

- Для первой буквы (А): $\mathbf{b_1 = (y^{k1} * M_1) \bmod p = (10^7*1) \bmod 23 = 14}$,
- Для второй буквы (б): $\mathbf{b_2 = (y^{k2} * M_2) \bmod p = (10^7*2) \bmod 23 = 05}$,
- Для третьей буквы (р): $\mathbf{b_3 = (y^{k3} * M_3) \bmod p = (10^7*18) \bmod 23 = 22}$
- И т.д.

Шифртекст **C**= 17 14 17 05 17 22.....

Длина сообщения **удваивается (основной недостаток алгоритма)**

Расшифрование.

Используем (7): $\mathbf{M_1 = (b_1 ((a_1)^x)^{-1}) \bmod p = (14 ((17)^3)^{-1}) \bmod 23 =}$

$\mathbf{= (14 * (4913)^{-1}) \bmod 23 = ((14 \bmod 23)* (\underline{(4913)^{-1} \bmod 23}\,)) \bmod 23 =}$

$\mathbf{= (14*5) \bmod 23 = 70 \bmod 23 = 1}$, **т.е** $\mathbf{M_1}$ **= «A»**

Здесь $\mathbf{(4913)^{-1}) \bmod 23 = 5}$**,** т.к. 4913 * **5** mod 23 = 1

Или используем (8): $\mathbf{M_1 = (b_1 (a_1)^{p-1-x}) \bmod p =(14* (17)^{23-1-3}) \bmod 23=}$

$\mathbf{= (14* (17)^{19}) \bmod 23=(14* 239072435685151324847153) \bmod 23 =1}$

И.т.д.

!!! В схеме Эль-Гамаля необходимо использовать различные значения случайной величины **k** для зашифрования различных сообщений **M** и **M’**. Если использовать одинаковые **k**, то для соответствующих шифртекстов (**a**, **b**) и (**a’**, **b’**) выполняется соотношение $\mathbf{b (b')^{-1} = M (M')^{-1} (\bmod p)}$. Из этого выражения можно легко вычислить **M**, если известно **M’**.

УПРАВЛЕНИЕ КРИПТОГРАФИЧЕСКИМИ КЛЮЧАМИ

Управление ключами – информационный процесс, включающий в себя три элемента:

- генерацию ключей;
- накопление ключей;
- распределение ключей.

В серьезных ИС используются специальные аппаратные и программные методы генерации случайных ключей (датчики ПСЧ).

Под накоплением ключей понимается организация их хранения, учета и удаления.

Вся информация об используемых ключах должна храниться в зашифрованном виде. Ключи, зашифровывающие ключевую информацию, называются мастер-ключам (мастер-ключи каждый пользователь должен знать наизусть)

Распределение ключей

- Распределение ключей (РК) – самый ответственный процесс в управлении ключами. К нему предъявляются два требования:
- оперативность и точность распределения;
- скрытность распределяемых ключей**.**

Введем обозначения:

$\mathbf{I_A}$ – идентификатор стороны **A**;

$\mathbf{D_A}$ – секретное криптопреобразование стороны **A** (с использованием секретного ключа асимметричной криптосистемы);

$\mathbf{E_A}$ – открытое криптопреобразование стороны **A** (с использованием открытого ключа асимметричной криптосистемы);

$\mathbf{T_A}$ – временной штамп (метка) стороны **A**;

$\mathbf{R_A}$ – случайное число, выбранное стороной **A**.

РК на основе симметричных систем.

1.Получение двумя пользователями общего ключа от центрального органа – центра распределения ключей (ЦРК) или центра сертификации (ЦС).

Проблемы:

- Одно проникновение в систему злоумышл-ка компрометирует ЦРК;
- ЦРК может долгое время участвовать в пассивном подслушивании, прежде чем это будет обнаружено.

Возможные решения:

- Иерархическая (древовидная) система с пользователями, находящимися на листьях, и ЦРК в промежуточных узлах.

2. Используя открытые сети.

Проблема: атака человек (встреча) посередине.

Возможное решение:

На основе протокола рукопожатия.

Алгоритм рукопожатия (основа протокола SSL)

А и В желают определить <u>общий секретный ключ </u>(**К**). Они <u>знают открытые ключи друг друга</u>.

1. **А** посылает **В** сообщение **$C= E_B (I_A , R_A)$**; **E_B** – процедура зашифрования с открытым ключом **В**, **I_A** – идентификатор **А** и **R_A** – случайное число.

2. **В** расшифровывает **С** и получает **I_A**и **R_A.**

В посылает **$C' = E_A (I_B , R_A)$** в адрес **А**;

После расшифрования **С‘** **А** может проверить, что **В** получил **R_A**, поскольку только **В** может расшифровать **С**.

3. **А** посылает **В** сообщение **$C'' = E_B(K_B)$,**

В расшифрует **С''** и сможет проверить что **А** получил

I_B, поскольку только **А** может расшифровать **С'**.

<u>Тем самым А и В **аутентифицировали друг друга**</u>.

Теперь **А** посылает **В** **$C''' = E_B(K)$, В** расшифровывает сообщение и получает **К**.

Алгоритм обеспечивает как секретность, так и аутентичность при обмене ключом К. Может исп-ся T_A – временной штамп (метка)

РК на основе асимметричных систем.

Пользователи **A** и **B** должны обменяться своими открытыми ключами.

Управление открытыми ключами может быть организовано с помощью оперативной или автономной службы

Проблемы:

Аутентичность. Если **A** думает, что $\mathbf{E_c}$ в действительности является $\mathbf{E_в}$, то **A** может зашифровать сообщение с помощью $\mathbf{E_c}$ и дать возможность **C** расшифровать сообщение, используя $\mathbf{D_c}$.

Целостность. Любая ошибка в передаче открытого ключа сделает его бесполезным.

Возможные решения:

Использование сертификатов

Использование сертификатов

- Обеспечивает одновременно аутентичность и целостность при распределении открытых ключей.
- Заключается в использовании сертификатов.
- Имеется центральный орган (ЦО, сертификационный центр), как и в случае распределения секретных ключей.
- Каждый пользователь может осуществлять безопасное взаимодействие с ЦО. Для этого требуется, чтобы у каждого пользователя был открытый ключ ЦО – $\mathbf{E_{ЦО}}$.
- Каждый пользователь **A** может зарегистрировать в ЦО свой открытый ключ $\mathbf{E_A}$. Поскольку $\mathbf{E_{ЦО}}$ является открытым, это можно сделать по почте, по открытому каналу электросвязи и т.п.
- При регистрации в ЦО **A** будет следовать определенной аутентификационной процедуре.

A получает сертификат, подписанный ЦО и содержащий $\mathbf{E_A}$, ЦО формирует сообщение **M**, содержащее $\mathbf{E_A}$, идентификационную информацию для **A** : $\mathbf{(I_A)}$, период действия сертификата и т.п.

- ЦО вычисляет $\mathbf{CERT_A = D_{ЦО}(M)}$, который и становится сертификатом **A.**

$\mathbf{CERT_A}$ делается общедоступным документом, который содержит $\mathbf{E_A}$ и аутентифицирует его, поскольку сертификат подписан ЦО.