Лекция 3

Криптосистема РША и анализ ее стойкости

1. Математический базис криптосистемы РША и других криптосистем

Модульная арифметика

a:n= { c, res (a) n≠0

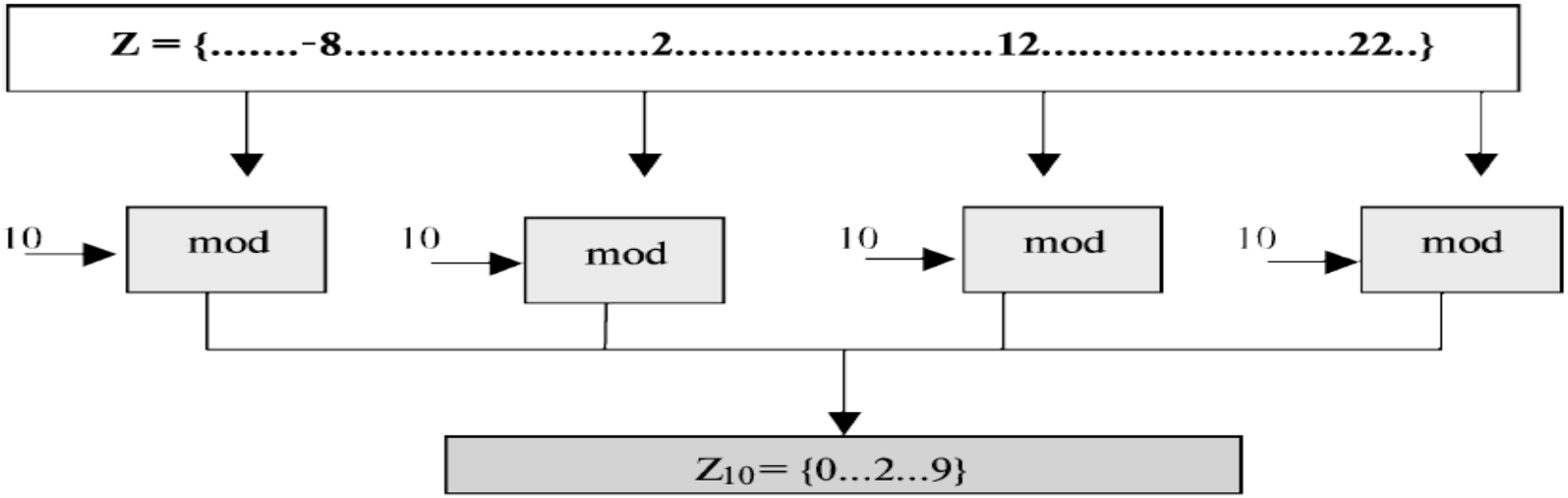
a=cn+res(a) - уравнение деления

Остаток от деления

***amodn* = r**

Результат операции по модулю n – целое **положительное** число меньшее чем n. Операция по модулю создает набор чисел Z_n, который в модульной арифметике называется **системой наименьших вычетов по модулю n**.

Пример множества Z_{10} . Z_{10}={0,1,2,3,4,5,6,7,8,9}.



Два числа a и b называются *сравнимыми по modn*, если их разность a-b делится на n.
Обозначение сравнимых чисел a≡b (modn)
Оператор сравнения отображает элемент Z на элемент Zn.

Например, числа -8, 2, 12, 22 сравнимы по модулю 10.
Сравнимые по модулю числа принадлежат **одному классу вычетов.**

Принято класс вычетов обозначать ***наименьшим положительным числом в этом классе***

В модульной арифметике определены три бинарных операции: сложение, вычитание, умножение.

Переместительный закон (коммутативный)
$(a+b)\bmod n=(b+a)\bmod n$

Сочетальный закон (ассоциативный)
$(a+(b+c))\bmod n=((a+b)+c)\bmod n$

Распределительный закон

$a(b+c)\bmod n=(ab)\bmod n+(ac)\bmod n$

$(a+b)\bmod n=(a(\bmod n)+b(\bmod n))\bmod n$

$(ab)\bmod n=(a\bmod n)(b\bmod n)\bmod n$

$-\mathbf{a\bmod n=(n-a)\bmod n}$

Разложение числа на множители

$$n = p_1^{a_1} p_2^{a_2} \cdots p_s^{a_s}$$

p_i - различные простые числа

a_i - положительные целые числа

Пример:

$n=90=2 \cdot 3^2 \cdot 5$

Число p называется простым, если оно не имеет делителей кроме тривиальных (1,-1, p, -p).

Наибольший общий делитель

Наибольшим общим делителем (НОД) двух чисел v и u называется наибольшее целое число, которое делит оба числа.

Нахождение НОД:

- Прямой метод - разложение чисел на множители.

  u=210 = 2•3•5•7, v=135=3•3•3•5. НОД(210,135)=15

- **Алгоритм Евклида.**

$u=a_1v+b_1$ 210=1*135+75

$v=a_2b_1+b_2$ 135=1*75+60

…

$b_{k-3}=a_{k-1}b_{k-2}+b_{k-1}$ 75=1*60+**15**

$b_{k-2}=a_kb_{k-1}+b_k$ 60=4*15+0

Если b_k=1, то НОД(u,v)=1,
если b_k=0, то НОД(u,v)= b_{k-1}

Евклид –древнегреческий математик (365-300 г до н.э.)

Числа u и v называются взаимопростыми, если НОД(u,v)=1

Теоремы Эйлера и Ферма

Функция Эйлера $\varphi(x)$. Определяет число натуральных чисел меньших *x* и взаимно простых с *x*. $\varphi(1)=1$.

$\varphi(6)=2$. $\varphi(xy)=\varphi(x)\bullet\varphi(y)$. $\varphi(p)=p-1$, если p простое.

Теорема Эйлера:

Если a и m взаимно простые числа, то

$a^{\varphi(m)}=1(modm)$ *Пример: a=5, m=6*

$5^2(mod6)=25(mod6)=1$

Эйлер Леонард 1707-1783 г.
1731-1741, 1761 работал в Петербургской АН

Теорема Ферма:

Если p простое число и p не делит a, то

$a^{p-1}=1(modp)$ *Пример: a=2, p=7* $2^6(mod7)=64(mod7)=1$

Ферма Франсуа
французский математик
1601-1665 г

Обращение элемента по модулю n

Обратный элемент **x** к числу **a** - это такое целое число, которое удовлетворяет сравнению

$$xa \equiv 1 (mod n).$$

Обозначение обратного элемента - a^{-1}.

Обратный элемент существует только тогда, когда НОД(a,n)=1.

Пример. n=7,a=5. $a^{-1}=3$. $5 \bullet 3=15$, $15(mod 7)=1$.

Нахождение обратного элемента:

Известно **каноническое** представление НОД в виде

$$НОД(a,n)=z_1 a+z_2 n,$$

где z1,z2 –целые не обязательно положительные числа.

Так как для взаимопростых чисел a и n НОД(a,n)=1, то

$1= (z_1 a+z_2 n) mod n=(z_1 a) mod n$. Следовательно,

$a^{-1} = z_1 \, mod n$

Пример нахождения обратного элемента

n=17, a=13, a^{-1}=?

1. **и**используя алгоритм Евклида, находим НОД(17,13),

17=1*13+4

13=3*4+1, НОД=1

2. Найдем числа z_1 и z_2, удовлетворяющие условию:

1= (z1*13+z2*17)mod17=z1*13(mod17).

1=13-3*4

4=17-1*13

1=13-3*4=13-3*(17-1*13)=4*13-3*17

z1=4, z2=-3

3. a^{-1}= **z1=4**

4. Проверка (4*13)mod17=52mod17=1

Пример 2

1. **и**спользуя алгоритм Евклида, находим НОД(97,77),

97=1*77+20
77=3*20+17
20=1*17+3
17=5*3+2
3=2*1+1, НОД=1

1=3-2*1 → 3-(17-5*3)*1=6*3-17*1=
2=17-5*3 =6*(20-1*17)-17*1=6*20-7*17=
3=20-1*17 =6*20-7*(77-3*20)=27*20-7*77=
17=77-3*20 =27*(97-77)-7*77=27*97-34*77
20=97-1*77

Получили представление
1=(z1*97+z2*77)mod97=(z2*77)mod97

a^{-1}= **z2=-34mod97= 63**

Проверка (63*77)mod97=4851mod97=1

Второй способ нахождения обратного элемента

Если p –простое число, то на основании теоремы Ферма $1=a^{p-1}(\mathrm{mod} p)$.

Умножая обе части равенства на a^{-1} получаем правило вычисления обратного элемента:

$$a^{-1}=a^{p-2}(\mathrm{mod} p)$$

Возведение в степень

$y=a^x(\mathrm{mod} n)$

Прямой способ: $(a\cdot a\cdot \ldots a\cdot a)\mathrm{mod} n$

1. Быстрый способ возведения: **Пример:** $3^{37}\mathrm{mod}7$

1a) Представим показатель степени в двоичном виде
$x=2^{k-1}x_{k-1}+2^kx_k+\ldots+2^2x_2+2^1x_1+2^0x_0$
$37=32+4+1=100101$

1b) Найдем k степеней основания *a* путем последовательного возведения в квадрат a. $3^1=3\mathrm{mod}7$, $3^2=2\mathrm{mod}7$, $3^4=4\mathrm{mod}7$, $3^8=2\mathrm{mod}7$, $3^{16}=4\mathrm{mod}7$, $3^{32}=2\mathrm{mod}7$.

1c) Перемножим между собой только те степени, которым соответствуют ненулевые коэффициенты в двоичном представлении числа x. То есть это, когда $x_i = 2^0, 2^2, 2^5$
$3^1=3\mathrm{mod}7$, $3^4=4\mathrm{mod}7$, $3^{32}=2\mathrm{mod}7$.
$y=(2\cdot 4\cdot 3)\mathrm{mod}7=3$.

Возведение в степень

2. Быстрый способ возведения Д.Кнута.

- **представим показатель степени в двоичном виде;**
- **каждую единице заменим парой букв КУ (квадрат+умножение);**
- **каждый ноль заменим буквой К (квадрат);**
- **в образовавшейся последовательности вычеркнем первую пару КУ;**
- **над основанием a проводим вычисления, согласно полученной последовательности.**

Пример: $3^{37}(mod7)$

37 = 100101 = ~~К~~УККККУККУ= КККУККУ

$3 \to 3^2 mod7=2 \to 2^2 mod7=4 \to 4^2 mod7=2 \to (2\cdot 3) mod7=6 \to 6^2 mod7= 1 \to 1^2 mod7= 1 \to (1\cdot 3) mod7=3$

Сложность вычислений для операции возведения в степень: $N \cong O(2\log x)$.

2. Принцип построения КС РША

Формирование пар открытых/закрытых ключей для КС РША

Каждый пользователь КС РША, допустим A, выполняет следующие операции для формирования пары ключей:

1) генерирует пару простых чисел p и q;

2) вычисляет $n = p \cdot q$ и функцию Эйлера $\varphi(n) = (p-1)\cdot(q-1)$;

3) генерирует e, где $1 \le e \le \varphi$, такое что $\gcd(e;\varphi) = 1$

4) находит число $d = e^{-1} \bmod \varphi$, т. е. решение уравнения $e \cdot d = 1 \bmod \phi(n)$

5) выбирает числа e, n как свой открытый ключ, а d – как свой секретный ключ.

Описанные выше операции могут быть выполнены достаточно быстро даже для чисел n, имеющих 100 и более десятичных разрядов.

- *Действительно, операции выполняются следующим образом:*
- 1) случайным генерированием чисел и тестированием Миллера–Рабина;
- 2) обычным умножением, что требует $O(\log p^2)$ битовых операций;
- 3) использованием алгоритма Евклида;
- 4) при помощи расширенного алгоритма Евклида для нахождения обратного элемента.
-
- Вся процедура генерирования пар ключей выполняется один раз на длительное время действия этих ключей.

Шифрование в КС РША

- Предположим, что пользователь *B* хочет передать пользователю *A* сообщение *M* в зашифрованном виде. Для этого он выполняет следующие шаги:
- 1) получает подлинный открытый ключ пользователя *A* (e_A , n_A);
- 2) преобразует сообщение *M* в последовательность целых чисел M_1 , M_2 , … , M_i , … , не превосходящих $(n_A - 1)$;
- 3) для каждого числа M_i , соответствующего части сообщения, формирует криптограмму c_i по правилу $c_i = M_i^{e_A} \bmod n_A$ и отправляет результат пользователю *A*.
- Видно, что процедура шифрования может быть выполнена достаточно быстро при использовании алгоритма быстрого возведения в степень по модулю.

Дешифрование в КС РША

Пользователь A для дешифрования предназначенной ему криптограммы c_i выполняет следующие операции с использованием своего секретного ключа d_A:

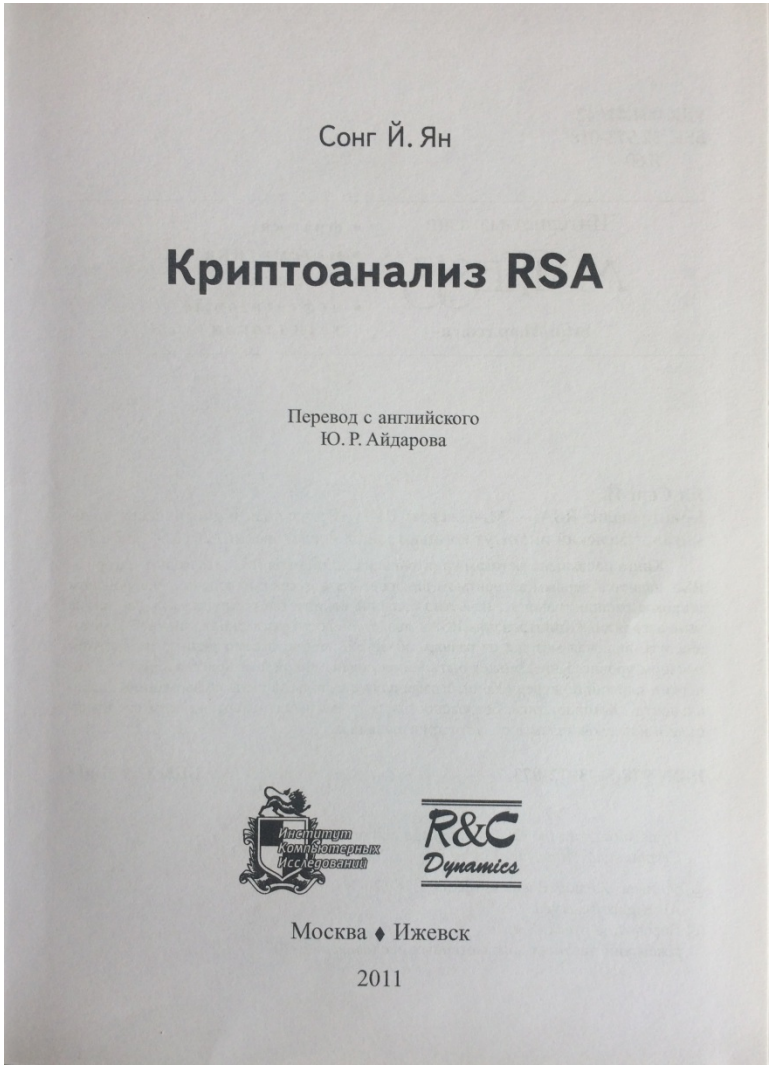
а) дешифрует $M_i = c_i^{\,d_A} \bmod n_A$, $i = 1, 2, \ldots$;

б) преобразует число M_i в содержательный вид (форму представления информации).

Видно, что операция дешифрования выполняется достаточно быстро.

- *Докажем, что представленная выше процедура дешифрования действительно позволяет получить истинное сообщение* $M = M_i$, *которое и было ранее зашифровано.* $c^d \bmod n = (M^e)^d \bmod n = M$
- **Доказательство.** Поскольку по условию $e \cdot d = 1 \bmod \varphi$, то существует такое число «*k*», что $e \cdot d = 1 + k \cdot \varphi$. Предположим сначала, что $gcd(M, p) = 1$, тогда по теореме Ферма имеем
- $$M^{p-1} = 1 \bmod p . \qquad (3.1)$$
- Возведем обе части (3.1) в степень $k(q-1)$, а затем умножим обе части на *M*:
- $$M^{1+k\,(p-1)(q-1)} = M \bmod p . \qquad (3.2)$$
- Рассмотрим показатель степени в левой части равенства (3.2):
- $$1 + k(p-1)(q-1) = 1 + k \cdot \varphi = e \cdot d$$
- Подставляя это выражение в (3.2), получим $M^{1+k\,(p-1)(q-1)} = M^{e \cdot d}$. Из равенства (3.2) получаем, что $M^{e \cdot d} = M \bmod p$, но, с другой стороны, $M^e = c$, следовательно, $c^d = M \bmod p$, что и требовалось доказать.

3. Анализ стойкости КС РША



Основные атаки на КС РША

1. Атака факторизации n

Для КС РША имеем следующие соотношения, связывающие ключи, сообщение и криптограмму:

$$c = M^e \bmod n\ ;$$

$$M = c^d \bmod n\ ;$$

$$n = p \cdot q; \qquad (3.3)$$

$$e \cdot d = 1 \bmod \varphi\ ;$$

$$\varphi = (p-1)\cdot(q-1).$$

Из уравнений (3.3) видно, что система РША может быть вскрыта, если удастся найти p и q, т. е. факторизовать n.

Исходя из этого факта p и q должны выбираться такой большой разрядности, чтобы факторизация числа n потребовала необозримо большого времени, даже с использованием всех доступных и современных средств вычислительной техники.

- В настоящее время задача факторизации чисел не имеет полиномиального решения. Разработаны лишь некоторые алгоритмы, упрощающие факторизацию, но их выполнение для факторизуемых чисел большой разрядности все равно требует необозримо большого времени. Действительно, сложность решения задачи факторизации для наилучшего известного сейчас алгоритма факторизации равна $O\left(e^{\sqrt{\ln(n)\cdot\ln\ln(n)}}\right)$ [3]. Так, например ln n = 200, если , то число операций будет приблизительно равно $1{,}37 \cdot 10^{14}$.

  При увеличении числа разрядов n до 1000 и более время факторизации становится совершенно необозримым.

Если теперь предположить, что алгоритм факторизации неизвестен, но удалось как-то в полиномиальное время найти секретный ключ d, то можно доказать, что число n тогда можно факторизовать, т. е. найти такие p и q за полиномиальное время, что $n = p \cdot q$ [3]. Если каким-то другим способом удалось в полиномиальное время найти параметр φ , то ясно, что КС РША может быть вскрыта.

Знание φ позволяет просто факторизовать n, и, следовательно, задача нахождения φ вычислительно эквивалентна задаче факторизации. (стр. 150).

Утверждение. (Полезное для анализа стойкости криптосистем с открытым ключом.) *Пусть* $n = p \cdot q$, *где* $p \cdot q$ *– простые числа.* $p \neq q$. *Тогда числа* p *и* q *можно найти, если известно* n *и* $\varphi(n) = (p-1)\cdot(q-1)$

□

□ **Доказательство.** Будем рассматривать p, q как пару неизвестных целых чисел, для которых задано их произведение $p \cdot q = n$ и известна сумма, поскольку $p + q = n + 1 - \varphi(n) = p \cdot q + 1 - (p-1)\cdot(q-1) = 2b$, где b – некоторое целое число.

□

□ Два числа, сумма которых равна 2b , а произведение равно n, являются очевидно корнями уравнения $x^2 - 2bx + n = 0$ (теорема Виета). Тогда корни квадратного уравнения и есть необходимые числа p и q:

□ .

$$p = b + \sqrt{b^2 + n^2};\ \ q = b - \sqrt{b^2 + n^2}$$

□

□ Сложность решения этого уравнения – $O(\log^3 n)$

- Тем не менее для систем РША нельзя строго утверждать, что φ стойкость этой КС эквивалентна задаче факторизации [3]. Подобное свойство имеет место для иных КС, например для КС Рабина, которая будет рассматриваться далее.
- **2. Атака дискетного логарифмирования d**
- Другой естественной атакой на КС РША является дискретное логарифмирование. Эта атака (при известном сообщении)
- $$M = c^d \bmod n$$
- выполняется следующим образом:
- $$d = \log_c M \bmod n.$$
- Однако задача дискретного логарифмирования по модулю многоразрядных чисел также относится к трудным в математике, и оказывается, что она имеет почти такую же сложность, как и задача факторизации [3].

Пример атаки дискретного логарифмирования

Пусть известен открытый ключ и шифртекст (e,n,c)=(17,2773,2258)

Найдем открытый текст, используя логарифмирование.

1. Выберем любую часть открытого текста M так, чтобы НОД(n,M)=1, например, M=2113 и вычислим

c=M^e =2113^17=340 (mod2773).

2. Вычисли d, взяв дискретный логарифм M с основанием c по modn.

$d=\log_{340} 2113=137$ (mod2773)

3. Теперь с помощью d можно расшифровать c=2258

M=c^d=2258^137=1225 (mod2773)

4. Результат легко проверить

c=M^e=1235^17=2258 (mod2773)

Замечание! Найденное d может отличаться от истинного d=e^-1 modφ

3. Побочные атаки на КС РША

Помимо двух основных атак на КС РША – *факторизации* и *логарифмирования* – существует ряд побочных атак, основанных на предположении об использовании в этой КС таких параметров или режимов, которые значительно упрощают ее криптоанализ.

Рассмотрим далее некоторые из этих атак, связанных с неправильным выбором параметров КС РША при ее разработке или эксплуатации.

Первая побочная атака.

Выбор малой величины открытого ключа e.

- Смысл такого выбора *e* для КС РША состоит в том, что это позволяет ускорить процедуру шифрования. Рассмотрим в качестве примера величину *e = 3*. Предположим, что с таким малым *e* производится шифрование в КС РША и одно и то же сообщение посылается хотя бы трем разным пользователям, имеющим разные *n*.

Покажем, что в этом случае можно восстановить сообщение *M* без знания ключа.

Последовательность криптограмм будет выглядеть следующим образом:

$$C_1 = M^3 \bmod n_1$$
$$C_2 = M^3 \bmod n_2; \qquad (3.4)$$
$$C_3 = M^3 \bmod n_3.$$

Весьма вероятно, что будет выполнено условие gcd (n_i , n_j) =1 при $i \neq j$. Вспомним китайскую теорему об остатках, которая гласит, что система уравнений (3.4) имеет общее решение: $x = M^3 \bmod (n_1 \cdot n_2 \cdot n_3)$, которое может быть найдено в полиномиальное время.

Вероятно также, что сообщение $M < n_1 \cdot n_2 \cdot n_3$, и тогда дешифрование выполняется тривиальным образом как извлечение кубичного корня из числа: $M = \sqrt[3]{x}$

Для защиты от этой атаки необходимо либо не использовать малые *e*, либо не отправлять одни и те же сообщения разным пользователям. Если в этом есть объективная необходимость, то сообщение нужно немного изменить, добавляя, например, в конце его небольшие различные случайные числа. Такой метод называется методом «*подсаливания*» сообщения.

Вторая побочная атака.
Атака при малом объеме возможных сообщений.

Предположим, что количество сообщений ограничено значениями $M_1, M_2, \ldots, M_r$, где r обозримо. (Это могут быть, например, различные команды – вперед, назад, влево, вправо и т. п.). Тогда сообщение может быть легко расшифровано.

Действительно, пусть криптограмма C перехвачена. Тогда необходимо попытаться зашифровать все команды известным открытым ключом и определить ту криптограмму, которая совпадает с принятой C:

$$\left.\begin{array}{l} C_1 = M_1^e \bmod n; \\ C_2 = M_2^e \bmod n; \\ \vdots \\ C_r = M_r^e \bmod n \end{array}\right\} \stackrel{?}{=} C.$$

Способ борьбы с такой атакой – это «*подсаливание*» сообщений (т. е. присоединение к ним небольших цепочек бит, полученных с использованием чисто случайного датчика).

Третья побочная атака. Малая экспонента дешифрования (d) (Математические основы)

На основе алгоритма Евклида можно рациональное число представить в виде цепной дроби

$$a/b = q_0 + \cfrac{1}{q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots\, q_{n-1} + \cfrac{1}{q_n}}}}$$

$$\frac{803}{154} = 5 + \cfrac{1}{4 + \cfrac{1}{1 + \cfrac{1}{2}}}$$

Цепная дробь, полученная из $[q_0, q_1, q_2, \cdots q_{n-1}, q_n]$ отбрасыванием всех элементов после некоторого номера k, называется k-й подходящей дробью

Атака Винера (Wiener)

При выборе малого d существенно упрощается алгоритм дешифрования («малая» в данном случае означает, что $d \approx \sqrt{N}$) . Эта атака основывается на следующей

Теореме.

Пусть p, q – простые числа, N=pq - модуль; d – секретный ключ; e – ключ шифрования; $ed = 1 \bmod \varphi(N)$ и пусть

$$\begin{cases} q < p < 2q \\ d < \frac{1}{3}\sqrt[4]{N} \end{cases}$$

Тогда значение d может быть легко вычислено как знаменатель одной из подходящих дробей k/d – разложения e/N в цепную дробь.

$$\left| \frac{e}{N} - \frac{k}{d} \right| < \frac{1}{2d^2}$$

Пусть N=160523347, e=60728973

$$\frac{e}{N} = 0 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{4 + \cfrac{1}{12 + \cfrac{1}{102 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{3 + \cfrac{1}{2 + \cfrac{1}{2 + \frac{1}{36}}}}}}}}}}}}}}$$

$$= [0, 2, 1, 1, 1, 4, 12, 102, 1, 1, 2, 3, 2, 2, 36]$$

Получаем следующие подходящие дроби

$$\left[0, \frac{1}{2}, \frac{1}{3}, \frac{2}{5}, \frac{3}{8}, \frac{14}{37}, \frac{171}{452}, \frac{17456}{46141}, \frac{17627}{46593}, \frac{35083}{92734}, \frac{87793}{232061}, \frac{298462}{788917}, \frac{684717}{1809895}, \frac{1667896}{4408707}, \frac{60728973}{160523347}\right].$$

Вспомним, что $ed = 1 \bmod \varphi(n)$, то есть $ed = k\varphi(n) + 1$ Откуда следует, что

перебором, для каждой k/d подходящей дроби, вычисляется $\varphi(N) = (ed - 1)/k$ и решается квадратное уравнение

$$p^2 - (N - \varphi(N) + 1)p + N = 0$$

Проверяется, является ли p простым множителем.

Предположим k/d=14/37, тогда

$$\phi(N) = \frac{ed - 1}{k} = \frac{60728973 \cdot 37 - 1}{14} = 160498000$$

$$p^2 - (N - \phi(N) + 1)p + N = 0$$

$$\Longrightarrow p = 12347, \; q = N/p = 13001.$$

Значения p и q найдены успешно, т.к.

$$N = pq = 12347 \cdot 13001 = 160523347.$$

Четвертая побочная атака. Атака с использованием мультипликативного свойства шифра РША.

- Из описания КС РША следует, что для любых сообщений M_1 , M_2

$$(M_1 \cdot M_2)^e = M_1^e \cdot M_2^e = C_1 \cdot C_2 \bmod n$$

- где $C_1 = M_1^e \bmod n$, $C_2 = M_2^e \bmod n$. Это свойство может использовать злоумышленник. Предположим, что злоумышленник *E* хочет дешифровать сообщение *C*, предназначенное для пользователя *A*, и предположим, что *A* согласен дешифровать любую другую криптограмму для *E*, помимо *C*.

Тогда *E* может дешифровать *C*, действуя следующим образом:

1) *E* выбирает $x \in Z_n$ и вычисляет $\tilde{C} = C \cdot x^{e_A} \bmod n$. Затем *E* просит *A* дешифровать $\tilde{C}$;

2) *A* выполняет эту просьбу – дешифрует: $\tilde{M} = \tilde{C}^{d} \bmod n$, затем передает результат злоумышленнику *E*. Поскольку

$$\tilde{M} = \tilde{C}^{d_A} = C^{d_A} \underbrace{\left(x^{e_A}\right)^{d_A}}_{x} \bmod n = Mx \bmod n$$

то E может выполнить следующий шаг;

3) *E*, зная $\tilde{M}$, определяет $M = \tilde{M} \cdot x^{-1} \bmod n$, т. е. дешифрует криптограмму *C*.

- Чтобы защититься от такой атаки, **нельзя дешифровать чужие сообщения.** Если все же от этого нельзя отказаться, то после дешифрования надо проверить, что получилось – случайный или осмысленный текст. Если получился случайный текст, то не передавать по запросу дешифрованное сообщение *E*.

Пятая побочная атака.
Атака на систему РША, использующую общие модули для нескольких пользователей.

Это типичная ситуация при генерировании пар ключей для пользователей некоторым общим для них «центром распределения ключей» (рис. 1).

Рис. 1. Атака на КС РША, использующую общие модули

- В этом случае любой пользователь i, имеющий ключи (e_i, d_i), способен определить любую другую пару ключей. Действительно, знание своего секретного ключа d_i позволяет ему факторизовать n, т. е. определить q и p .

$$ed = 1 \bmod \varphi(n) \qquad ed = k\varphi(n) + 1$$

- (детали см. стр 160 уч. пособия)

- Зная e_j (как открытый ключ другого, j-го пользователя), а также используя тот факт, что $e_j \cdot d_j = 1 \bmod (p-1)(q-1)$ i -й пользователь может вычислить секретный ключ d_j любого другого пользователя.

Шестая побочная атака.
Циклическая атака.

Предположим, что известна лишь одна криптограмма C, полученная в криптосистеме РША. Тогда злоумышленник может легко найти ее преобразования:

$$C_1 = C^{e^1} \bmod n; C_2 = C^{e^2} \bmod n; C_3 = C^{e^3} \bmod n, ..., C_r = C^{e^r} \bmod n$$

Эти вычисления он продолжает до тех пор, пока результат не совпадет с исходной криптограммой C. (Это событие должно произойти рано или поздно на каком-то шаге k, так как шифрование – это по существу перестановка чисел $\{0, 1, 2, \ldots, n-1\}$.) Тогда он может найти сообщение как $M = C^{e^{k-1}} \bmod n$, поскольку $M^e = C^{e^k} = C$.

Однако в [3] доказывается, что такой метод дает и факторизацию числа n, и поэтому при больших n этот подход не лучше прямого метода факторизации модуля КС РША.

Седьмая побочная атака. Отсутствие шифрования.

- Этот случай возможен, если в результате шифрования получаем открытое сообщение, т. е. $M^e \bmod n = M$. Такое условие должно выполниться хотя бы для одного из сообщений, например, для сообщений $M = 0, 1, n - 1$. На самом деле таких сообщений, которые *вообще!* не шифруются [3], существует в точности $[1 + \gcd(e - 1, p - 1)][1 + \gcd(e - 1, q - 1)]$. Их число всегда не менее 9. Однако при случайном выборе q и p доля таких сообщений будет ничтожно мала и они почти никогда не встретятся на практике.

- **Физические атаки на КС РША**
  - *Атака, основанная на нахождении времени выполнения определенных операций*
- Такие атаки предполагают, что шифрование/дешифрование выполняются в аппаратной форме. Пусть существует некоторый невскрываемый чип дешифрования (рис. 2).

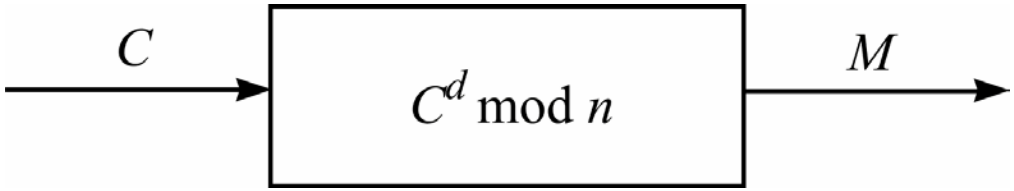


Рис. 2. Дешифрование при помощи невскрываемого чипа

Напомним, что быстрое возведение в степень состоит в последовательном возведении криптограммы в квадрат и умножениях на криптограмму по модулю n.

- **Пример.** Рассмотрим дешифрование при помощи возведения в степень:

$$C^{171} = \left(\left(\left(\left(\left(\left((C)^2\right)^2 C\right)^2\right)^2 C\right)^2\right)^2 C\right)^2 C$$

- что соответствует выполнению следующих операций возведения в квадрат и умножения: (ККУККУККУКУ)
- кв. кв. умнож. кв. кв. умнож. кв. кв. умнож. кв. умнож.
-
- Типично то, что операция умножения требует больше времени, чем возведение в квадрат. Если злоумышленник может определить время выполнения последовательности операций при дешифровании в легальном чипе, например, контролируя потребление энергии или излучения в окружающее пространство, то он может восстановить и последовательность действий, а следовательно, и секретный ключ *d*.

- (Защитой от такой атаки может быть зашумление окружающего пространства или источников питания специальными шумовыми генераторами или нормализация времени выполнения всех операций.)

- *Атака внешним воздействием.*

- Эта атака выполняется при помощи радиационного или электромагнитного излучения.
- В некоторых случаях для сокращения времени производят вычисления $M = C^d \bmod N = C^d \bmod pq$
- по составляющим модуля с последующим использованием китайской теоремы об остатках, т. е. $M_1 = C^d \bmod p$, $M_2 = C^d \bmod q$, а затем находят $M = (M_1 \cdot a + M_2 \cdot b) \bmod n$ при выполнении условий: $a = 1 \bmod p$ и $a = 0 \bmod q$; $b = 0 \bmod p$ и $b = 1 \bmod q$. (Время вычислений сокращается в 4 раза).

- Предположим, что в момент вычисления M_1 внутри чипа все происходит правильно, а при вычислении M_2 возникает ошибка (скажем, за счет созданного злоумышленником электромагнитного импульса). Тогда $M_1 = C^d \bmod p$, $M_2 \neq C^d \bmod q$, и результат вычисления сообщения будет содержать ошибки, т. е.
- $\tilde{M} = \left(a \cdot M_1 + b \cdot \tilde{M}_2\right) \bmod n$
- Далее злоумышленник, зная M и $\tilde{M}$, находит:

$$\left(M - \tilde{M}\right) \bmod p = a \cdot M_1 - a \cdot M_1 = 0 \bmod p \quad \textit{или} \left(M - \tilde{M}\right) \bmod p = kp$$

$$\left(M - \tilde{M}\right) \bmod q = b \cdot M_2 - b \cdot \tilde{M}_2 = b \cdot \left(M_2 - \tilde{M}_2\right) \bmod q \neq 0 \bmod q$$

- Отсюда следует, что нахождение $\gcd\left(M - \tilde{M}, n\right) = \gcd\left(kp, pq\right) = p$
- дает нетривиальную факторизацию n, т. е. его сомножитель p, поскольку $M - \tilde{M}$ делится на p, но не делится на q, т. е. $\gcd(M - \tilde{M}, n) \neq n$ (если бы делилось, то НОД =n и факторизации бы не было)

Выбор параметров для КС РША

Как было отмечено ранее, правильный выбор параметров и режимов работы может обеспечить стойкость КС РША для любых существующих вычислительных средств криптоанализа.

При этом необходимо обратить внимание прежде всего на выбор величины модуля n и его множителей:

1) уместно выбрать битовую длину $l(n)$ модуля n более 768 (а для обеспечения высокой стойкости $l(n) \geq 1024$);

2) для того чтобы избежать атак с использованием некоторых специальных алгоритмов факторизации, числа p и q должны быть примерно одинаковой величины, т. е. порядка $\sqrt{n}$ (512 бит для высокой стойкости), причем их разность $(p - q)$ не должна быть малой, поскольку тогда эту разность можно будет найти перебором и затем факторизовать n.

- Иногда рекомендуется использовать так называемые *строго простые числа* p и q, которые удовлетворяют следующим условиям:
- $p - 1$ и $q - 1$ имеют большой простой делитель r;
- $p + 1$ и $q + 1$ имеют большой простой делитель x;
- $r - 1$ имеет большой простой делитель y.
- Однако последние исследования показывают [3], что эти условия не являются необходимыми для обеспечения стойкости даже относительно всех атак при выборе $l(n) > 1024$;
- 3) выбор малых экспонент шифрования e (число $e = 3$ используется на практике, так как это требует одного возведения в квадрат и одного умножения), вообще говоря, допустим, но для защиты от соответствующей атаки необходимо тогда «подсаливать» сообщения.

- Используют также экспоненту шифрования $e = 2^{16} + 1 = 65573$, что требует 16 возведений в квадрат и одного умножения. Данный метод, даже без «подсаливания» сообщений, имеет преимущество перед выбором экспоненты $e = 3$, поскольку используемая для малых экспонент атака будет эффективной, если только *одно и то же сообщение* шифруется и посылается одновременно 65537 пользователям (!), что, конечно, маловероятно;

- 4) при построении КС для группы пользователей, использующих шифры РША, в случае когда именно центры распределения ключей снабжают их ключевыми данными, необходимо избегать распределения общих модулей.