Грушо А.А., Применко Э.А., Тимонина Е.Е.

Анализ и синтез криптоалгоритмов

КУРС ЛЕКЦИЙ

Москва
2000

Введение

Курс лекций составлен в соответствии с программой одноименного курса, читаемого в течение ряда лет на факультете защиты информации в Российском государственном гуманитарном университете.

Защита информации опирается на набор механизмов и методов, которые, будучи разумно использованы, позволяют гарантировать невозможность атак противника на защищаемую информацию. Криптоалгоритмы являются существенной частью этого набора. Криптоалгоритмы – это алгоритмы преобразования данных, использующие “секрет”. Основной параметр качества криптоалгоритма – устойчивость к попыткам противника открыть “секрет”. Такая устойчивость в криптографии называется стойкостью. Криптографическую стойкость надо обосновывать, так как в защите критической информации логика: “я не могу раскрыть “секрет”, следовательно, никто не может” неприменима. Методы обоснования криптографической стойкости основаны на накопленном опыте раскрытия “секретов” криптоалгоритмов. Этот опыт сформулирован в методах анализа шифров (один из классов криптоалгоритмов), т.е. в найденных методах и подходах в раскрытии “секретов”. Могут найтись гении, придумавшие новые методы анализа, тогда надо пересмотреть стойкость уже используемых криптоалгоритмов или опереться на вновь найденные методы при синтезе новых криптоалгоритмов. Кроме стойкости криптоалгоритмы должны удовлетворять ряду других требований, например, экономичности, скорости реализации и т.п. Совокупность подходов, позволяющих удовлетворить заданному набору требований, составляет теорию синтеза криптоалгоритмов. Методы анализа и синтеза основаны на математике. В основном нужды анализа и синтеза криптоалгоритмов обслуживают дискретная математика, алгебра и теория вероятностей. Более того, криптография дала развитие многим направлениям математики. Изучение вопросов анализа и синтеза начнем с описания простейших шифров.

В соответствии с традицией современной криптографии курс лекций содержит описание наиболее известных универсальных методов криптоанализа, методов анализа блочных и поточных шифров, методов анализа хэш-функций и алгоритмов с несимметричным ключом. По мере знакомства с методами анализа читателю предлагаются разделы, содержащие методы синтеза криптоалгоритмов.

Глава 1. Примеры шифров.

1.1. Определение шифра, простейшие примеры.

Пусть даны конечные множества *X, Y, K*. Будем интерпретировать элементы *X* как открытые сообщения, элементы Y как шифрованные тексты, элементы *K* - как ключи.

Определение 1. Отображение $T: X \times K \to Y$ называется шифром, если для $\forall k \in K \ \exists T^{-1}(y, k) = x$.

Пример 1. Пусть $A = \{a_1,...,a_m\}$ - конечный алфавит, S_m - множество всех подстановок на *A*. Для некоторого натурального *n* проложим $X = A^n$. Если $x = (a_{i_1},...,a_{i_n})$, $k \in S_m$, то определим шифр простой замены следующим образом:

$T(x, k) = (k(a_{i_1}), k(a_{i_2}),..., k(a_{i_n}))$.

Так как в группе S_m для каждого элемента *k* есть обратный k^{-1} так, что $k\,k^{-1} = e$ - тождественная подстановка, то

$T^{-1}((k(a_{i_1}), k(a_{i_2}),..., k(a_{i_n})), k) =$

$= (k^{-1}k(a_{i_1}), k^{-1}k(a_{i_2}),..., k^{-1}k(a_{i_n})) = (a_{i_1},...,a_{i_n}) = x.$

Таким образом, мы доказали, что для $\forall$ *n* шифр простой замены действительно удовлетворяет определению шифра.

Пример 2. Пусть $A = \{a_1,...,a_m\}$ - конечный алфавит, *n* -натуральное, S_n - симметричная группа подстановок на множестве $\{1,...,n\}$, $X = A^n$. Если $x = (a_{i_1},...,a_{i_n})$, $k = \begin{pmatrix} 1 \cdots & n \\ j_1 \cdots & j_n \end{pmatrix} \in S_n$, то шифр перестановки на *X* определяется следующим образом:

$T(x, k) = (a_{i_{j1}}, a_{i_{j2}},..., a_{i_{jn}})$.

Если $k^{-1}(j_1,...,j_n)$ - обратная к *k* подстановка в S_n, то

$T^{-1}((a_{i_{j1}}, a_{i_{j2}},..., a_{i_{jn}}), k) = (a_{i_1},...,a_{i_n})$.

Тем самым также доказано, что шифр перестановки удовлетворяет определению шифра.

Пример 3. В отечественной криптографии следующий шифр Вернама [Ш., с.346] получил название гаммирование. Пусть $A = \{0, ..., m\text{-}1\}$ - алфавит, $X = A^n$- множество открытых текстов. Рассмотрим кольцо вычетов *Z/m*. Положим $K = A^n$ и для $\forall x \in X, k \in K$ определим преобразование

$y = T(x, k) = x + k$,

где + - сложение по mod m, т.е. сложение в Z/m (соответственно обратная операция обозначается -). Введенное преобразование - шифр, т.к. для любого фиксированного k существует обратное отображение

$T^{-1}(y, k) = y - k = x$.

Традиционно для данного вида шифра ключ K обозначается греческой буквой γ, откуда в отечественной криптографии и появилось соответствующее название. Так как гамму при больших n часто записывали в блокнот, то за рубежом используется термин блокнотный шифр (pad).

Пример 4. Американский стандарт шифрования, известный под названием DES [Хоф., с.250]. Этот шифр реализует простую замену на алфавите из всех двоичных векторов длины 64 бита. Выбор подстановки осуществляется по ключу длиной 56 бит. Ключ преобразуется в 16 48-битовых комбинаций. Подстановки вычисляются для любого 64-битного вектора открытого текста путем реализации 16 циклов повторения одного преобразования. На вход i-го преобразования поступают два 32-битных (левый L_{i-1}, правый - R_{i-1}) вектора, полученных от предыдущего цикла или это входной вектор переставленный по перестановке IP, и разделенный пополам на две 32-битных части (левую L_0 и правую R_0). Еще одним входом для i-го цикла преобразования является 48-битная комбинация ключа. Результатом i-го цикла преобразования является составленный из двух 32-битных частей (левой L_i и правой R_i) 64-битный вектор промежуточного шифртекста. Схема DES условно представлена на рисунке 1. На этой схеме $\oplus$ - побитовое сложение двух 32-битных векторов по модулю 2, IP - перестановка бит, IP^{-1} - обратная к IP перестановка. Перестановки IP и IP^{-1} представлены в таблице 1.

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP $^{-1}$							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Таблица 1. Начальная и обратная перестановки DES.

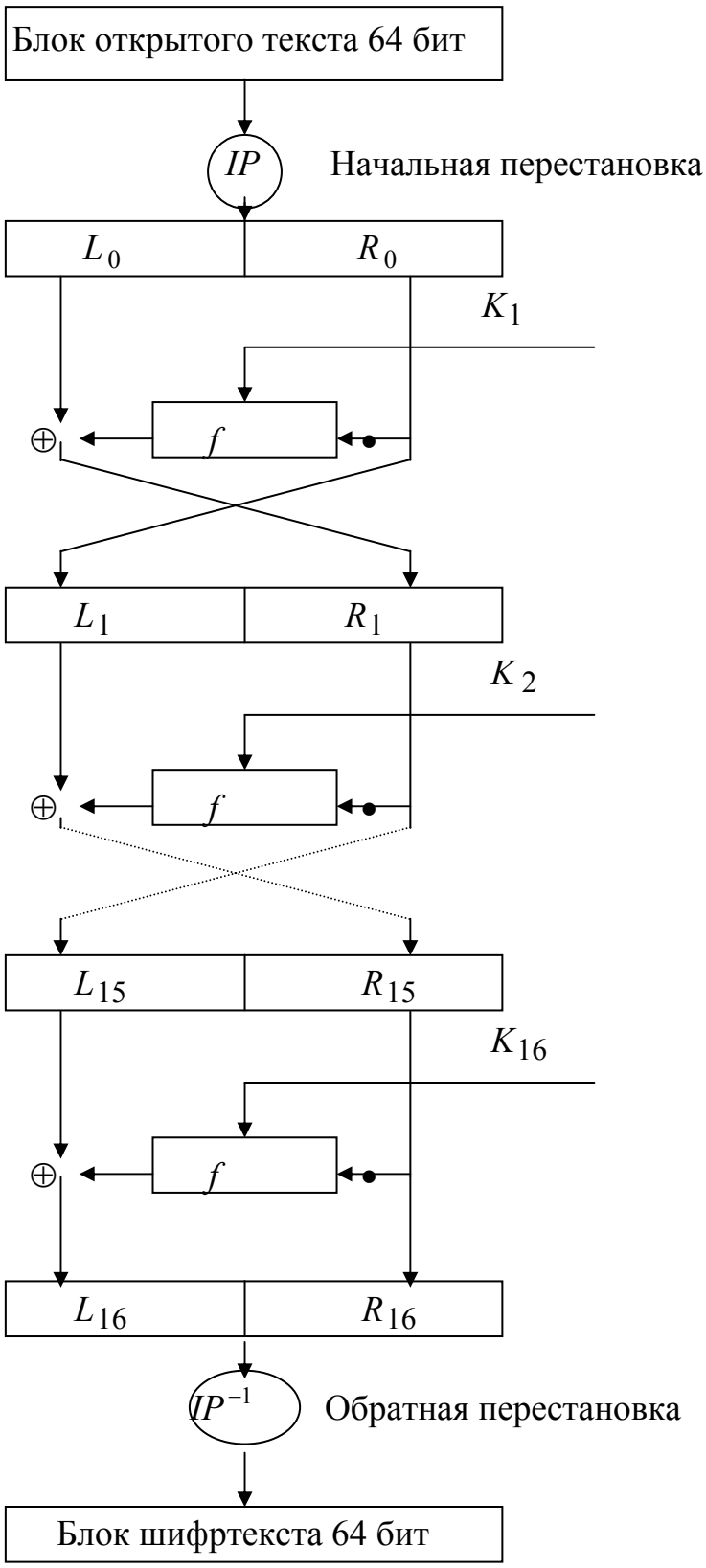


Рис. 1. Схема алгоритма DES.

Алгоритм вычисления функции *f* представлен на рисунке 2.

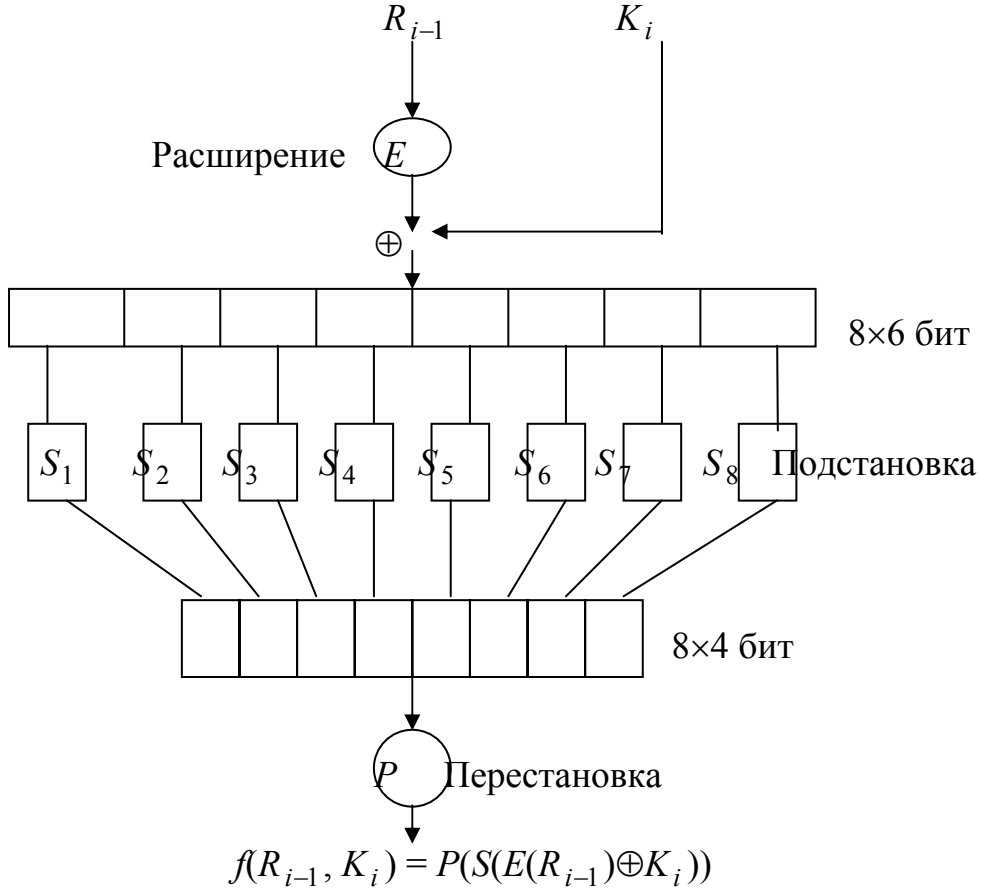


Рис. 2. Схема функции усложнения DES.

E							***P***			
32	1	2	3	4	5		16	7	20	21
4	5	6	7	8	9		29	12	28	17
8	9	10	11	12	13		1	15	23	26
12	13	14	15	16	17		5	18	31	10
16	17	18	19	20	21		2	8	24	14
20	21	22	23	24	25		32	27	3	9
24	25	26	27	28	29		19	13	30	6
28	29	30	31	32	1		22	11	4	25

Таблица 2. Функция расширения и перестановка в функции f усложнения DES.

Схема S_i бокса представлена на рисунке 3. *S* боксы DES приведены в таблице 3. Здесь контрольные биты CL и CR являются входами в таблицу по горизонтали, а 1 – 4 биты – входами в таблицу по горизонтали. Так, например, $S_1(1,0,1,1,1,0)= (1,0,1,1)$.

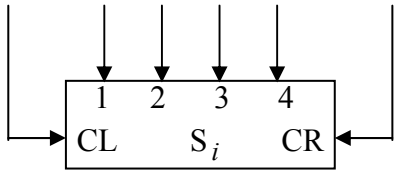


Рис. 3. Схема S-бокса DES. Здесь CL – левый контрольный бит, CR – правый контрольный бит.

S_1																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	**1**	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1	**0**	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
1	**1**	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
0	**1**	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	2
1	**0**	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
1	**1**	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
0	**1**	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
1	**0**	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	**1**	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
0	**1**	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
1	**0**	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
1	**1**	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
0	**1**	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
1	**0**	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
1	**1**	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**

0	**0**	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
0	**1**	10	12	4	2	7	12	9	5	6	1	13	14	0	11	3	8
1	**0**	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
1	**1**	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
0	**1**	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	**0**	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
1	**1**	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																	
CL	**CR**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	**0**	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
0	**1**	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
1	**0**	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
1	**1**	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Таблица 3. S боксы DES.

Ключи K_1,..., K_{16} получаются из 64 бит ключа DES следующим образом. Определим v_i, $1 \le i \le 16$ следующим образом: v_i=1 для i = {1, 2, 9, 16} и v_i=2 для остальных i. Из 64 бит ключа выбираются 56 бит и делятся на две части C_0 и D_0 по 28 бит согласно PC1 в таблице 4. C_0 и D_0 записываются в два циклических регистра сдвига влево. На каждом цикле оба регистра сдвигаются на v_i, $1 \le i \le 16$, бит, получая таким образом C_i и D_i. В сумме получается сдвиг на 28 бит, что возвращает заполнения регистров в исходное положение. Ключ K_i, $1 \le i \le 16$, получается из C_i и D_i согласно PC2 в таблице 4.

***PC*1**													
C_i							D_i						
57	49	41	33	25	17	9	63	55	47	39	31	23	15
1	58	50	42	34	26	18	7	62	54	46	38	30	22
10	2	59	51	43	35	27	14	6	61	53	45	37	29
19	11	3	60	52	44	36	21	13	5	28	20	12	4

***PC*2**					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2

41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Таблица 4.

При любом данном k расшифрование DES отличается от зашифрования только тем, что регистры сдвигаются вправо в порядке, обратном зашифрованию. Начинается расшифрование с перестановки бит IP^{-1}, а заканчивается перестановкой IP. DES сыграл в США огромную роль при формировании системы электронных платежей. В настоящее время известно более 65 официальных производителей реализаций DES в виде программного обеспечения или микросхем.

Пример 5. Отечественный стандарт шифрования ГОСТ 28147-89 [Schn. 96, с. 331]. Рассмотрим режим простой замены ГОСТ 28147-89.Этот режим является основным и на его основе реализуются другие режимы шифрования.

Исходный открытый текст M (бинарная последовательность) разбивается на блоки длиной 64 бит:

$M = M_1 M_2 ... M_t$, $|M_i| = 64$.

Длина ключа K равна 256 битам. Ключ K разбивается на блоки длиной 32 бит:

$K = P_1 P_2 ... P_8$, $|P_i| = 32$, $i = 1,\ldots,8$.

При шифровании используется ключевая последовательность:

$(K_1,K_2,...,K_{31},K_{32})=(P_1,P_2,...,P_8, P_1,P_2,...,P_8, P_1,P_2,...,P_8, P_8,P_7,...,P_1)$,

а при расшифровании используется ключевая последовательность:

$(K_1',K_2',...,K_{31}',K_{32}')=(P_1,...,P_7,P_8, P_8,P_7,...,P_1, P_8,P_7,...,P_1, P_8,P_7,...,P_1)$,

Таким образом,

$K_i' = K_{n+1-i}$, $i = 1,2,...,n$; $n = 32$.

Функция сдвига R определяется следующим образом:

$R(\alpha_1\ \alpha_2 ...\ \alpha_{32}) = \alpha_{12}\ \alpha_{13} ...\ \alpha_{32}\ \alpha_1\ \alpha_2 ...\ \alpha_{11}$.

Обозначим через $+$ сложение по модулю 2^{32}; $\oplus$ - побитное сложение (исключающее ИЛИ); $+'$ - сложение по модулю $2^{32}-1$.

Блок подстановки реализует функцию:

$f_S(w)$, $|w| = 32$.

Функция $f_S(w)$ - это подстановка на множестве $\{0, 1\}^{32}$. Параметр S является сменным и определяет долговременные ключи криптосистемы ГОСТ 28147-89. Подстановка S задается в виде последовательности $S = (\pi_1, \pi_2, ..., \pi_8)$, где π_i - подстановка на множестве $\Omega = \{0, 1, 2, ..., 9, A, B, C, D, E, F\}$ (множество всех цифр 16-ричной системы счисления). Каждый элемент из Ω естественным образом отождествляем с соответст-

вующим четырехразрядным двоичным числом (последовательностью из четырех бит). Из определения S следует, что число различных выборов этого параметра равно $(16)!^8$. Значение функции $f_S(w)$ вычисляется следующим образом. Если $w = v_1 v_2 \ldots v_8$, $|v_i| = 4$, то
$f_S(w) = \pi_1(v_1)\ \pi_2(v_2) \ldots \pi_8(v_8)$.

Процедура зашифрования выглядит следующим образом. Пусть $M = \alpha_1 \alpha_2 \ldots \alpha_{64}$ - блок открытого текста. Введем обозначения:

$a_0 = \alpha_{32}\ \alpha_{31} \ldots \alpha_1,\quad b_0 = \alpha_{64}\ \alpha_{63} \ldots \alpha_{33}$.

Тогда, процедура зашифрования блока M при помощи криптоалгоритма ГОСТ 28147 - 89 описывается следующим рекурсивными уравнениями:

$$\begin{aligned} x_i &= a_{i-1} + k_i, \\ y_i &= f_S(x_i), \\ z_i &= R(y_i), \\ a_i &= Z_i \oplus b_{i-1}, \\ b_i &= a_{i-1}, \qquad i = 1,\ldots,n\text{-}1;\ n\text{=}32. \end{aligned} \tag{1}$$

$$\begin{aligned} x_n &= a_{n-1} + k_n, \\ y_n &= f_s(x_n), \\ z_n &= R(y_n), \\ a_n &= a_{n-1}, \\ b_n &= z_n \oplus b_{n-1}. \end{aligned} \tag{2}$$

Если $a_n = \beta_1,\beta_2,\ldots\beta_{32}$, $b_n = \gamma_1,\gamma_2,\ldots\gamma_{32}$ и T - блок шифрованного текста, соответствующий блоку открытого текста M, то

$T = \text{ГОСТ}_K(M) = \beta_{32}\ldots\beta_1\ \gamma_{32}\ldots\gamma_1$.

Убедимся, что $\text{ГОСТ}_{K^|}(T) = M$, где $T = \text{ГОСТ}_K(M) = \beta_{32}\ldots\beta_1\ \gamma_{32}\ldots\gamma_1$,

$a_n = \beta_1\ldots\beta_{32}$, $b_n = \gamma_1\ldots\gamma_{32}$, $K^| = K_n\ K_{n-1}\ \ldots\ K_1$, т.е. $K_i^| = K_{n+1-i}$, $i = 1,\ldots,n$.

Из сказанного выше следует, что

$a_0' = a_n$, $b_0' = b_n$.

И при $i = 1$, из системы (1) получим:

$x_1' = a_0' + k_1' = a_n + k_n$.

Поскольку, согласно (2), $a_n = a_{n-1}$, то

$x_1' = a_{n-1} + k_n = x_n$.

Поэтому, учитывая (1) и (2) будем иметь:

$$\begin{aligned} y_1' &= f_S(x_1') = f_S(x_n) = y_n, \\ z_1' &= R(y_1') = R(y_n) = z_n, \\ a_1' &= z_1' \oplus b_0' = z_n \oplus b_n = z_n \oplus (z_n \oplus b_{n-1}) = b_{n-1}, \\ b_1' &= a_0' = a_n = a_{n-1}. \end{aligned}$$

Допустим, что мы уже установили справедливость соотношений

$$a_{i-1}' = b_{n-i+1},\quad b_{i-1}' = a_{n-i+1}, \tag{3}$$

при всех $k \le i$-1 , $i \ge 2$. Докажем их справедливость для $k = i$. Согласно алгоритму зашифрования и индуктивному предположению (3), из (1) следует, что

$$x_i' = a'_{i-1} + k'_i = b_{n+1-i} + k_{n+1-i} = a_{n+i} + k_{n+1-i} = x_{n+1-i}\ .$$

Следовательно,

$$y_i' = f_S(x_{n+1-i}) = y_{n+1-i},$$

$$z'_i = R(y_{n+1-i}) = z_{n+1-i},$$

$$a'_i = z'_i \oplus b'_{i-1} = z_{n+1-i} \oplus a_{n+1-i} = z_{n+1-i} \oplus (z_{n+1-i} \oplus b_{n-i}) = b_{n-i},$$

$$b'_i = a'_{i-1} = b_{n+1-i} = a_{n-i}.$$

Таким образом, для любого $i = 1,\ldots,n$.

$$a'_i = b_{n-i},\ b'_i = a_{n-i}.$$

В частности, $a'_{n-1} = b_1,\ b'_{n-1} = a_1$. Поэтому из (1) и (2) будем иметь:

$$x'_n = a'_{n-1} + k'_n = b_1 + k_1 = a_0 + k_1 = x_1,$$

$$y'_n = f_{s'}(x_1) = y_1; z'_n = R(y_1) = z_1,$$

$$a'_n = a'_{n-1} = b_1,$$

$$b'_n = z'_n \oplus b'_{n-1} = z_1 \oplus a_1 \oplus z_1 \oplus (z_1 \oplus b_0) = b_0.$$

Следовательно, T' = ГОСТ$_K(T) = M$ и взаимная однозначность криптографического преобразования ГОСТ 28147-89 установлена.

1.2. Стойкость шифров.

Данное в предыдущем параграфе определение шифра не разграничивает хорошие и плохие шифры. Однако главная цель шифра - надежная защита скрываемой информации. Поэтому качество шифра оценивается прежде всего понятием надежности защиты. В криптографии принят термин “стойкость” шифра. Под этим понимается устойчивость защиты перед атаками противника, получившего какие-либо сведения, связанные с шифром или информацией, закрываемой с помощью шифра, или просто сам шифртекст.

Рассмотрим подробнее ситуацию, в которой ведется криптоанализ. В определение шифра входят следующие параметры T, x, y, k, T^{-1}. Защищаемая информация описывается параметром x. Сама идея криптоанализа основана на том, что противник перехватил и имеет на руках y. Классифицируем основные условия криптоанализа.

1. Известны только один или несколько шифртекстов y. В этих условиях решают следующие задачи:

- найти T (определение типа шифра)
- найти T, T^{-1}, x (дешифрование по шифртексту).

2. Известны одна или несколько пар (x, y). В этих условиях надо определить вид шифра T (T^{-1}) и найти k. Это редкая ситуация.

3. Известны вид шифра T (T^{-1}), один или несколько шифртекстов y. Найти:

- x (бесключевое чтение);
- k, x (дешифрование по шифртексту при известной шифрсистеме).

4. Известны: вид шифра T, T^{-1}, одна или несколько пар (x, y). Найти k. Это типичные условия криптоанализа. Стойкость шифров оценивается именно в этих условиях. Иногда отдельно возникают ситуации, когда известно много пар (x, y) так, что можно подобрать x или y, удовлетворяющие некоторым дополнительным условиям. Тогда говорят об атаке с использованием выбранного открытого или шифрованного текстов.

5. Известны T, T^{-1}, шифртекст y или пары (x, y), некоторая форма преобразования $T(\ .\ , k)$, но неизвестны k и $T^{-1}(\ .\ , k)$. Допустимость таких постановок задач рассматривалась Диффи и Хеллманом в 1976 г. Системы шифрования, где допускается возможность знания $T(\ .\ , k)$ без раскрытия k и T^{-1} получили название систем с открытым ключом.

Разрешимость для противника указанных задач определяет стойкость шифра в соответствующих условиях, а методы решения этих задач называются методами криптоанализа. Универсальные методы криптоанализа - это такие методы, которые с определенными вариациями применимы ко многим шифрам.

В криптографии разработаны два базовых подхода в оценке стойкости шифров. Основы первого подхода (совершенная секретность) К. Шеннон изложил в своей работе [Ш., с.360] в 1948 г. Рассмотрим этот подход. К. Шеннон предложил следующую модель обращения с параметрами, описывающую шифрование и действия противника. Пусть $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1,\ldots,y_m\}$ - множества открытых сообщений и возможных шифртекстов. Открытый текст для передачи в шифрованном виде выбирается случайно в соответствии с распределением

$\{P(x_1), ...,P(x_n), P(x_i)\geq 0, \sum_{i=1}^{n} P(x_i) = 1\}$. Тогда при решении задачи вскрытия x_i по известному y_j противник может вычислить апостериорные вероятности сообщений, которые могли быть посланы $P(x_s \mid y_j)$, s=1,...*n*. Если задача дешифрования решена и открытый текст найден, то апостериорное распределение вырождено:

$P(x_i \mid y_j) = 1, P(x_s \mid y_j) = 0$ при $s \neq i$.

Таким образом в апостериорных вероятностях $\{P(x \mid y), x \in X\}$ отражаются даже частичные сведения об открытом тексте, полученном при перехвате криптограммы. Шеннон определил совершенную секретность (стойкость) шифра условием: апостериорные распределения на открытых текстах при любом $y \in Y$ совпадают с априорным распределением на *X*, т.е. $P(x \mid y) = P(x)$ для любых $x \in X,\ y \in Y$.

Пример 1. Пусть $X = \{0, 1\}^r$, $Y = \{0, 1\}^r$, $K = \{0, 1\}^r$, *r* - натуральное и

$T(x, k) = (x \oplus k)(\mathrm{mod}\ 2)$, $T^{-1}(y, k) = (y \oplus k)(\mathrm{mod}\ 2)$,

т.е. рассматривается гаммирование в двоичном алфавите. Пусть также $(P_X(x), x \in X)$ - произвольное распределение вероятностей на *X*. Предположим, что *k* выбирается случайно и равновероятно из *K*, то есть $\forall\ к \in K$, $P_K(k) = \frac{1}{2^r}$. Иными словами, знаки гаммы выбираются из множества $\{0, 1\}$ независимо друг от друга и от открытого текста с равными вероятностями $P(0) = P(1) = \frac{1}{2}$. Тогда для любого $y \in Y$

$$P(y) = \sum_{\substack{x\in X \\ k\in K \\ y=(x\oplus k)(mod\,2)}} P(x,k) = \sum_{x\in X} P_X(x) \sum_{\substack{k\in K \\ y=(x\oplus k)(mod\,2)}} P(k|x) =$$

$$= \sum_{x\in X} P_X(x)\frac{1}{2^r} = \frac{1}{2^r},$$

$$P(y \mid x) = \frac{P(x,y)}{P(x)} = \frac{P_X(x)P_K((x\oplus k)(mod\,2))}{P_X(x)} = \frac{1}{2^r}.$$

По формуле Байеса

$$P(x \mid y) = \frac{P_X(x)P(y|x)}{P(y)} = \frac{P_X(x)}{\frac{1}{2^r}2^r} = P_X(x).$$

Следовательно, гаммирование при помощи последовательности, полученной при помощи реализации независимых равновероятных случай-

ных векторов, является совершенным шифром.

Подход Шеннона превратил криптографию из искусства в науку, т.к. появилась возможность доказывать защищенность информации при помощи шифра. Однако реализовать последовательность независимых и равновероятных случайных величин оказалось делом сложным. Кроме того, объем ключа в совершенном шифре совпадает с объемом сообщения, что затрудняет выполнение условия секретности ключа.

Следующим шагом в теоретическом подходе к оценке стойкости шифров послужила оценка числа открытых текстов, которые мы можем получить, применяя все возможные способы преобразования имеющихся шифр текстов при неизвестном заранее ключе. Смысл этого подхода поясним на примере, не связанном с криптографией.

Пример 2. Пусть мы играем в игру: надо угадать слово по имеющимся в порядке их следования некоторым буквам этого слова. Предположим, что дана буква “п”. Если нет каких-либо ограничений на возможное слово, позволяющее однозначно назвать допустимое слово по одной известной из него букве, то существует много слов, имеющих букву “п”, и мы не можем даже приблизительно указать слово, которое нам предложили угадать. То есть мы не получаем достаточно информации, спрятанной за этим словом. В аналогичных условиях мы не можем восстановить слово, даже зная несколько букв. Например, нам известно, что первыми тремя буквами слова являются буквы “при”. Без дополнительных ограничений мы можем указать множество различных по смыслу слов, имеющих приставку “при”. Поэтому это буквосочетание почти ничто не дает к пониманию смысла неизвестного слова.

Аналогично, если при дешифровании уравнение $T(x, k) = y$ имеет много решений x, а ключ может принимать любые допустимые значения, то возможно много осмысленных текстов x, удовлетворяющих имеющимся в нашем распоряжении ограничениям и имеющих примерно одинаковую вероятность их появления в качестве открытых текстов. Тогда нет алгоритма, решающего задачу дешифрования (в смысле однозначного нахождения открытого текста и ключа). Наоборот, во многих моделях шифров можно доказать, что такое решение единственно.

Другой подход к определению стойкости берет начало в той же работе Шеннона [Ш., с.387] и называется практической стойкостью или сложностный подход к стойкости. Традиционно рассматриваются ситуации 3.б или 4. Рассмотрим выражение $T(x, k) = y$ как уравнение относительно x и k. Тогда решение этого уравнения предполагает существование алгоритма, для которого в математике определено понятие сложности [Гэ.]. Сложность характеризуется двумя параметрами: число операций для вычисления результата (трудоемкость алгоритма) и объем необходимой памяти.

Число операций при данном уровне развития вычислительной техники связано со временем работы алгоритма, поэтому стойкость можно выразить в терминах времени работы алгоритма дешифрования. Естественное требование надежности шифра - высокая сложность всех возможных алгоритмов дешифрования.

Замечание 1. Возможно повышать сложность алгоритмов дешифрования, повышая сложность преобразования $T(x, \kappa) = y$. Однако, если сложность вычисления $T(x, k)$ и $T^{-1}(y, k)$ при известном k будут велики, то практически такой шифр трудно использовать. Поэтому наряду с требованием сложности решения уравнения $T(x, k) = y$ при неизвестном k, привлекают естественное требование простоты вычисления $T(x, k)$ и $T^{-1}(y, k)$ при известном k.

Замечание 2. Требование высокой сложности при всех возможных алгоритмах дешифрования неконструктивно, т.к. опирается на потенциальную возможность перечисления всех возможных алгоритмов дешифрования. Практически [Ru.] это требование заменяется на реализуемое условие высокой трудоемкости при всех известных создателям шифра методах дешифрования. Большинство универсальных методов опубликовано в литературе и мы будем рассматривать задачу синтеза шифров, стойких относительно универсальных методов.

Глава 2. Универсальные методы криптоанализа.

По умолчанию, если не оговаривается противное, будем считать, что задача дешифрования решается в условиях 4 предыдущего параграфа, когда известна одна или несколько пар (x, y) и известны T и T^{-1}, а неизвестен только ключ.

Несмотря на название, не все универсальные методы всегда применимы. Так как некоторые из них значительно снижают стойкость шифра, то создатели стараются делать так, чтобы универсальные методы были к их шифрам неприменимы. Вместе с тем, все множества X, Y, K конечны. Поэтому потенциально можно перебрать все их элементы и найти все решения уравнения $T(x, k) = y$. Следовательно, множество методов, применимых к данной шифрсистеме, непусто.

2.1. Метод полного перебора.

Рассмотрим уравнение относительно $k \in K$ при известной паре $(x, y), x \in X, y \in Y$:

$$T(x, k) = y. \qquad (1)$$

Пусть для простоты для $\forall$ пары (x, y) существует единственное k, удовлетворяющее (1). Упорядочим множество K в соответствии с заданным порядком и будем последовательно проверять ключи из K на предмет равенства в уравнении (1). Если считать проверку одного варианта ключа $k \in K$ в уравнении (1) за одну операцию, то полный перебор ключей потребует $|K|$ операций, где знаком $|\cdot|$ обозначается число элементов в множестве. Пусть ключ в схеме шифрования выбирается случайно и равновероятно из множества K. Тогда с вероятностью $\frac{1}{|K|}$ трудоемкость метода полного перебора равна 1. Это происходит в том случае, когда случайно выбран ключ, расположенный в нашем порядке на первом месте. Поэтому естественно в качестве оценки трудоемкости метода взять математическое ожидание (среднее) число шагов в переборе до попадания на использованный ключ. Найдем среднее число шагов в методе полного перебора, когда порядок фиксирован, а выбор ключа случаен и равновероятен.

Пусть случайная величина τ - число опробований включительно до момента обнаружения использованного ключа. При $i = 1,..., |K|$ случайные величины $\xi_i = 1$, если использованный ключ находится в порядке на месте i и $\xi_i = 0$ в противном случае. Тогда

$$E\tau = \sum_{i=1}^{|K|} i\, P(\xi_i = 1). \qquad (2)$$

Если считать, что все ключи расположены в установленном порядке, то процедуру равновероятного выбора ключа можно представлять как равновероятный выбор числа i в последовательности натуральных чисел $1,...,|K|$. Тогда $P(\xi_i = 1) = \frac{1}{|K|}$ для любого $i = 1,...,|K|$. Подставляя полученные значения в (2) получим

$$E\tau = \frac{1}{|K|}\sum_{i=1}^{|K|} i = \frac{|K|(|K|+1)}{|K|\cdot 2}.$$

При больших $|K|$ можно приблизительно считать $E\tau \approx \frac{|K|}{2}$.

Пример 1. В DES 56-битный ключ. Значит $|K| = 2^{56} \approx 10^{17}$. Средняя трудоемкость полного перебора $2^{55} \approx 0{,}5\cdot 10^{17}$.

Алгоритмы полного перебора допускают распараллеливание, что позволяет значительно ускорить нахождение ключа. Можно предложить два направления в организации параллельного вычисления ключа [Ама.].

Во-первых, построение конвейера. Пусть алгоритм проверки равенства $T(x, к) = y$ представим в виде детерминированной цепочки простейших действий (операций)

$O_1, O_2,..., O_N$.

Возьмем N процессоров $A_1,..., A_N$, зададим их порядок и положим, что i-ый процессор выполняет три одинаковые по времени операции: 1) прием данных от (i-1)-го процессора; 2) выполнение операции O_i; 3) передача данных следующему (i+1)-му процессору. Тогда конвейер из N последовательно соединенных, параллельно и синхронно работающих процессоров работает со скоростью v/3, где v- скорость выполнения одной операции процессором. Если мы со скоростью v/3 подаем на вход ключи $k \in K$, то с такой же скоростью будем получать результат на выходе. Постоянная для процессора задержка N тактов между моментом входа ключа k и ответом, является ли ключ k решением (1), мала по сравнению с $|K|$. Поэтому рассмотренный конвейер подтверждает допустимость предположения о том, что сложный алгоритм вычисления $T(x, к)$ и сравнение с y, можно считать элементарной операцией (на самом деле, в нашем примере требуется 3 элементарные операции независимо от сложности алгоритма проверки ключа при $|K| >> N$).

Второе направление распараллеливания состоит в том, что множество

натуральных чисел 1,2,...,$|K|$ разбивается на непересекающиеся подмножества B_1,..., B_R. В каждом из них может быть свой порядок. Система из R машин перебирает ключи так, что i-ая машина осуществляет перебор ключей из множества B_i, i = 1,...,R. Система прекращает работу, если одна из машин нашла ключ.

Пример 2. Если одна машина опробует один ключ за 10^{-6} сек, то для того, чтобы найти ключ DES полным перебором за 24 часа на R-машинной системе при $|B_1| = ... = |B_R|$ надо, чтобы

$$R = \frac{10^{17}}{2 \cdot 864 \cdot 10^8} = \frac{10^9}{2 \cdot 864} = 5{,}787 \cdot 10^5 \text{ машин.}$$

Если нас устраивает срок 100 суток, то таких машин надо всего 5787.

В последнее время широкое распространение получили глобальные сети ЭВМ. Можно сделать вирус, который выполняет следующую задачу: опробует ключи в свободное время процессора, начиная с некоторого номера; в случае положительного решения передает ответ и самоуничтожается, а также распространяет вирус, который всем другим работающим над опробованием вирусам приказывает самоуничтожиться.

Аналогичная идея распараллеливания предложена в “китайской лотерее” [Schn. 96]. В каждый телевизор или радиоприемник на внутреннем рынке Китая, вмонтирован чип, который принимает открытый и шифрованный текст и начинает опробование заданного участка ключей DES. Если ключ найден, то аппарат дает сигнал, а его владелец должен сообщить о сигнале в правительственный орган, там его ждет приз. Если правительству нужен очередной ключ, то передается в широковещательной передаче открытый и шифрованные тексты, по которым включаются чипы в телевизорах и радиоприемниках. По данным [Schn. 96] на 1995 год “китайская лотерея” при использовании чипа в 1 млн. операций в сек. могла позволить вскрыть ключ DES в Китае за 280 сек, в США - за 97 сек.

Самой большой сложностью в изложенном подходе является организация деления ключевого множества. Однако можно сделать старт с любого числа и случайно выбранного ключа. Время опробования увеличится, но схема значительно упростится. Рассмотрим задачу о среднем времени опробования N процессорами (машинами) ключей из множества K при старте каждого процессора со случайной точки при каждом очередном опробовании. Пусть все машины проработали τ шагов. Тогда сделано $N\ \tau$ опробований. Рассмотренная задача имеет аналог в задаче распределения частиц в классической задаче о размещениях [Кол. 76]. Предположим, что все машины работают синхронно. Обозначим число тактов опробования до первого обнаружения ключа через η.

Если частицы бросаются порциями по N штук (т.е. машины работают

синхронно), то вероятность того, что из комплекта с номером i ни одна частица не попадет в данную ячейку равна $q = (1 - \frac{1}{|K|})^N$.

Тогда вероятность того, что произойдет первое попадание в данную ячейку на комплекте с номером t, равна

$$P(\tau = t) = q^{t-1}(1-q) = (1 - (1 - \frac{1}{|K|})^N)^{t-1} \cdot (1 - \frac{1}{|K|})^N .$$

Среднее геометрического распределения равно

$$E\tau = \frac{1}{1-q} = \frac{1}{1 - (1 - \frac{1}{|K|})^N} .$$

Так как $N << |K|$, то

$$E\tau \approx \frac{1}{1 - 1 + \frac{N}{|K|}} = \frac{|K|}{N} .$$

Пример 2. При случайном старте каждого чипа при полном опробовании и при времени одного опробования на одном чипе 10^{-6} сек для определения ключа за 100 суток потребуется $N = 11574$ машин.

То есть случайный старт в два раза увеличивает необходимое количество процессоров при одном и том же времени ожидания результата.

2.2. Аналитический метод.

Пусть множество ключей K представимо в виде прямого произведения множеств

$K = K_1 \times K_2 \times ... \times K_r$.

Например, множество ключей DES $K = \{0, 1\}^{56}$. Пусть дешифровщик получил открытое сообщение $x=x_1x_2...x_s$ и шифрсообщение $y= y_1y_2...y_s$, где для простоты считаем, что x и y слова длины s в одном алфавите A, т.е. $x_i, y_i \in A$. Тогда можно составить систему уравнений шифрования, выписанную для T в каком-либо базисе операций:

$$\begin{aligned} y_1 &= f_1(x_1,...x_s, k_1,....,k_r) \\ &..... \\ y_s &= f_s(x_1,...x_s, k_1,....k_r) \end{aligned} \qquad (1)$$

В этих уравнениях при известных f, x, y остаются неизвестными только $k_1,....,k_r$.

Отметим, что $|K| = |K_1| \times |K_2| \times ... \times |K_r|$ и полный перебор требует $\frac{|K|}{2}$ опробований.

Анализ получившейся системы (1) может подсказать более быстрые способы ее решения, чем полный перебор.

Пример 1. Пусть система (1) может быть преобразована к виду:

$$\begin{aligned} g_1(x, y) &= h_1(x, y, k_1) \\ g_2(x, y) &= h_2(x, y, k_1, k_2) \\ &..... \\ g_r(x, y) &= h_r(x, y, k_1, k_2,..., k_r). \end{aligned} \qquad (2)$$

Тогда [Ш., стр. 397], опробуя $|K_1|$ вариантов часть ключа $k_1 \in K_1$ и используя для проверки правильности первое уравнение в (2), мы можем восстановить k_1. Подставив полученное значение во второе уравнение мы используем его для проверки правильности варианта для $k_2 \in K_2$, затратив при этом не более $|K_2|$ операций опробования. Найдя k_2, мы подставим его в третье уравнение и т.д. В результате весь ключ $k = (k_1, k_2,..., k_r)$ мы получим, проведя не более, чем $|K_1| + |K_2| + ... + |K_r|$ опробований. Это число, как правило, значительно меньше, чем $\frac{|K|}{2} = \frac{1}{2}(|K_1| \times |K_2| \times ... \times |K_r|)$. Например, при $|K_1| = |K_2|$, r=2, трудо-

емкость метода $\sim 2\sqrt{|K|}$. В этом случае для 56-битного ключа вместо $\frac{1}{2}2^{56} \approx \frac{1}{2}10^{17}$ достаточно выполнить $2 \cdot 2^{28} < 10^9$ операций.

Пример2. Значительное упрощение задачи дешифрования мы получаем, когда в системе (1) мы каким-либо образом можем выделить линейную подсистему уравнений. В этом случае можно применить известные методы решения систем линейных уравнений, которые по трудоемкости и памяти значительно меньше переборных методов. В самом деле, пусть в (1) выделена подсистема

$$g_i(x, y) = \sum_{j=1}^{r} b_{ij} k_j,\ i = 1,...,t.$$

Применим для ее решения метод Гаусса и найдем трудоемкость для случая, когда $g_i(\cdot) \in \{0, 1\}$ и линейная система рассматривается над GF(2).

Если система линейных уравнений с r неизвестными совместна и имеет вид

$$\sum_{j=1}^{r} b_{ij} k_j = c_i,\ i=1,...,r,$$

то ее решение методом Гаусса предполагает на первом этапе выделение строк с $b_{i1} = 1$ не более r операций, вычитание из первой строки с $b_{i1} = 1$ остальных строк с $b_{m1} = 1$ не более $(r+1)(r-1)$ операций. Итого получаем оценку r^2 операций на первом этапе. На втором этапе делаем то же самое с подсистемой

$$\sum_{j=2}^{r} b_{ij} k_j = c_i^{(1)}.$$

Оценка трудоемкости здесь $(r-1)^2$ операций. Продолжая этот алгоритм, в результате получим трудоемкость метода

$$\sum_{k=1}^{r} k^2 = \frac{r(r+1)(2r+1)}{6} \approx \frac{r^3}{3}.$$

Таким образом, нахождение 64-битного ключа методом полного перебора составило бы $2^{64} \approx 10^{19}$ операций, а, если задача сведена к решению линейной системы уравнений, то трудоемкость не превосходит $\frac{64^3}{3} \approx 8{,}7 \cdot 10^4$ операций.

Пример 3. В книге Гилла [Гилл] линейные рекуррентные последовательности над GF(2) рекомендуются в качестве хорошего шифра. Линейная рекуррента описывается законом

$$\gamma_{n+t} = \sum_{i=0}^{t-1} \alpha_i \gamma_{n+i}, \qquad n = 0, 1, \ldots, \tag{3}$$

где сложение и умножение осуществляется в GF(2).

Иногда говорят, что такие рекуррентные последовательности порождаются линейными регистрами сдвига (регистром сдвига с линейной обратной связью). Последовательность $\{\gamma_s, s = 0, 1,...\}$ может использоваться в шифре гаммирования по mod2, ключом является начальное заполнение регистра сдвига $(\gamma_0, \gamma_1, ..., \gamma_{t-1}) = \gamma(0)$. Нетрудно видеть, что существует матрица A такая, что любой вектор последовательных t значений рекурренты равен степени матрицы A, умноженной на $\gamma(0)$. Если закон рекурренты известен, то задача определения ключа сводится к решению системы линейных уравнений размера $t \times t$ и поэтому данный шифр нестойкий.

Если линейная рекуррента неизвестна, то при известной γ получаем линейную систему относительно коэффициентов α_i, решив ее, можем узнать ключ $\gamma(0)$ [Меу.].

Пример 4. Пусть двоичная гамма в шифре гаммирования по mod2 получена при помощи нелинейной рекурренты (при помощи регистра сдвига с нелинейной обратной связью или просто нелинейного регистра сдвига)

$$\gamma_{n+t} = f(\gamma_n, \gamma_{n+1}, \ldots, \gamma_{n+t-1}), \qquad n = 0, 1, \ldots \tag{4}$$

Пусть f такова, что при любом начальном заполнении регистра последовательность периодична, тогда порождающий ее нелинейный регистр сдвига называется регулярным. Тогда оказывается, что для любого начального заполнения существует линейная рекуррента вида (3) размера v ($v \geq t$) такая, что порождаемая ею последовательность совпадает с последовательностью, порождаемой при этом начальном заполнении регистром сдвига с нелинейной обратной связью.

Доказательство существования линейной рекурренты очевидно, так как, если последовательность $\gamma_0, \gamma_1, ..., \gamma_T, \gamma_{T+1}, ...$ периодична с периодом T, то линейный регистр сдвига вида $\gamma_{n+T} = \gamma_n$, n = 0, 1,..., порождает ту же последовательность, что (4).

Определение 1. Длина минимального линейного регистра сдвига, порождающего данную периодическую последовательность (регулярного нелинейного регистра сдвига), называется линейной сложностью регулярного нелинейного регистра сдвига.

Если гамма для шифра гаммирования порождается регулярным нелинейным регистром сдвига с линейной сложностью T, и ключом является начальное заполнение регистра, то из существования линейного регистра

сдвига длины T, порождающего данную гамму, следует, что нахождение ключа сводится к линейной системе размера T. Откуда сложность дешифрования не превосходит $\frac{T^3}{3}$. Однако построение эквивалентного минимального линейного регистра сдвига по последовательности гаммы, не является простой задачей. Вместе с тем, чем меньше линейная сложность, тем проще задача дешифрования. Для полноты картины необходимо найти конструктивные условия регулярности нелинейного регистра сдвига. Докажем следующую теорему.

Теорема 1. *Нелинейный регистр сдвига (4) над GF(2) регулярен тогда и только тогда, когда*

$$f(\gamma_n, \gamma_{n+1}, \ldots, \gamma_{n+t-1}) = \gamma_n + g(\gamma_{n+1}, \ldots, \gamma_{n+t-1}). \qquad (5)$$

Д о к а з а т е л ь с т в о. Если f имеет структуру (5), тогда для любой цепочки $x_2, \ldots, x_t$

$$f(0, x_2, \ldots, x_t) = 1 + f(1, x_2, \ldots, x_t). \qquad (6)$$

Обратно, если при всех $x_2, \ldots, x_t$ выполняется (6), то имеет место представление (5). В самом деле, обозначив через $g(x_2, \ldots, x_t) = f(0, x_2, \ldots, x_t)$, получим (5). Всегда из (6) следует, что

$$f(x_1, x_2, \ldots, x_t) = x_1 f(1, x^{3}, \ldots, x_t) + (1 + x_1) f(0, x_2, \ldots, x^{4}) =$$

$$= x_1 (f(0, x_2, \ldots, x_t) + f(1, x_2, \ldots, x_t)) + f(0, x_2, \ldots, x^{4}) =$$

$$= x_1 + f(0, x_2, \ldots, x_t).$$

Рекуррента определяет отображение множества t-мерных векторов в себя

$$F: (x_1, \ldots, x_t) \to (x_2, \ldots, x_t, f(x_1, \ldots, x_t)).$$

Рекуррента регулярна тогда и только тогда, когда F – взаимно-однозначное соответствие. Если верно (5) и существуют два вектора, переходящие при F в один, то они могут отличаться только в x_1 и

$$f(0, x_2, \ldots, x_t) = f(1, x_2, \ldots, x_t),$$

что противоречит (6). Обратно, при регулярности (4) имеем взаимно-однозначное отображение F, то есть для любых $(x_2, \ldots, x_t)$ вектора

$$(x_2, \ldots, x_t, f(0, x_2, \ldots, x_t)) \neq (x_2, \ldots, x_t, f(1, x_2, \ldots, x_t)).$$

Следовательно,

$f(0, x_2, ..., x_t) = 1 + f(1, x_2, ..., x_t)$,

что и требовалось доказать.

2.3. Метод "встреча по середине".

Рассмотрим каскадный метод построения сложного шифра из исходных простых. Даны два шифра $T_1(x, k_1)$ и $T_2(z, k_2)$. Шифртекст y получается из открытого текста x композицией отображений T_1 и T_2, то есть

$y = T_2(T_1(x, k_1), k_2)$.

Ключ шифра - вектор $k = (k_1, k_2)$, где $k_1 \in K_1$, $k_2 \in K_2$, $K = K_1 \times K_2$. Разумеется, в T_1 и T_2 согласованы области определения и значений при любых $k_1 \in K_1$ и $k_2 \in K_2$ и расшифрование происходит применением следующего преобразования:

$x = T_1^{-1}(T_2^{-1}(y, k_2), k_1)$.

Трудоемкость полного перебора равна $\frac{|K_1||K_2|}{2}$. Вместе с тем можно сократить перебор, увеличив используемую память [Men., с.235]. Пусть для известной пары (x, y) ключ (k_1, k_2) определяется единственным образом. Для всех $k_1^{(i)} \in K_1$, $i = 1, ...$, $|K_1|$, построим таблицу

$$\begin{aligned} z_1 &= T_1(x, k_1^{(1)}) \\ &..... \\ z_{|K_1|} &= T_1(x, k_1^{|K_1|}). \end{aligned} \qquad (1)$$

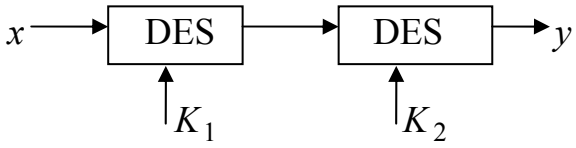
Вычисление таблицы (1) потребует $|K_1|$ операций опробования. Построим следующую таблицу для всех $k_2^{(j)} \in K_2$, $j=\overline{1,|K_2|}$.

$$\begin{aligned} z'_1 &= T_2^{-1}(y, k_2^{(1)}) \\ &..... \\ z'_{|K_2|} &= T_2^{-1}(y, k_2^{(|K_2|)}). \end{aligned} \qquad (2)$$

Вычисление (2) потребует $|K_2|$ операций опробования. Объединим таблицы (1) и (2) и проведем их упорядочение в соответствии с некоторым порядком на множестве значений промежуточных шифртекстов z. Известно [Ре.], что сложность упорядочения таблицы размера $(|K_1|+|K_2|)$ оценивается величиной $(|K_1|+|K_2|)\ln(|K_1|+|K_2|)$. Пара $z_i = z'_j$ при некоторых (i, j) определяет использованный ключ $(k_1^{(i)}, k_2^{(j)})$. Для ее нахо-

ждения достаточно просмотреть таблицу. Число операций, которые мы затратили $(|K_1|+|K_2|)+(|K_1|+|K_2|)\ \ln(|K_1|+|K_2|)$. Если $|K| = N$ и $|K_1|=|K_2|$, то метод "встреча по середине" дает оценку $\sqrt{N}\ \ln N$, что значительно меньше N при $N \to \infty$. Такой метод требует памяти размера $2N$.

Пример 1. Двойной DES. Пусть для повышения стойкости используется повторное шифрование алгоритмом DES на разных ключах K_1 и K_2.



Тогда метод "встреча посередине" дает оценку трудоемкости $2^{56}\, ln\, 2^{112} \approx 100 \cdot 10^{17} = 10^{19}$, что значительно меньше полного перебора ($\approx 10^{34}$). Однако требуемая память $\approx 10^{17}$ является значительной.

Пример 2. Рассмотрим следующий шифр, допускающий быструю реализацию на ЭВМ. Для произвольного целого неотрицательного числа n обозначим <n> - двоичное представление n, l<n> - длина двоичного представления n, [n] - разбиение двоичного числа n на байты (дописав слева нули, если необходимо). Для вектора целых неотрицательных чисел $\bar{n} = (n_1, n_2, ..., n_s)$ положим $<\bar{n}> = (<n_1>,< n_2>, ..., <n_s>)$, $[\bar{n}\ ]= ([n_1], [n_2], ...,[n_s])$.

Рассмотрим симметричную группу подстановок S_{256} и для любого $g \in S_{256}$ будем отождествлять записи:

$$g = \begin{pmatrix} ...m... \\ ...i_m... \end{pmatrix} = \begin{pmatrix} ... < m > ... \\ ... < i_m > ... \end{pmatrix} = \begin{pmatrix} ...[m]... \\ ...[i_m]... \end{pmatrix},\ m = 0,...,255.$$

Пусть дана таблица $\mathfrak{T}$ подстановок из S_{256}: g_0 , ..., g_{M-1}, где M= 1024. Эта таблица записана в памяти так, что для i = 0,...,M-1, подстановка g_i имеет адрес <i> длины l<i> = 10. Тогда адреса $\alpha = (k_1, ..., k_8)$ 8 подстановок задаются двоичным вектором $<\alpha>= (<k_1>, ..., <k_8>)$, $l<\alpha> = 80$. С другой стороны $[\alpha] = ([r_1], ...,[r_8])$ - вектор из 10 байт. Пусть открытый текст x представим в виде последовательности байт $x = (x_1,..., x_l)$. В качестве шифра рассмотрим простую замену байт открытого текста по подстановке $g \in S_{256}$, полученной по правилу:

$$g = g_{k_1} ... g_{k_8},$$

где $<\bar{k}\ >= (<k_1>, ..., <k_8>)$ - 80-битный ключ шифрования, который выбирается случайно и равновероятно, k_i = 0,....M-1, $l<k_i> = 10$, g_{k_i} взято из таблицы $\mathfrak{T}$. Пусть даны открытый текст x и шифртекст y. Рассмотрим атаку

“встреча по середине” при условии, что таблица подстановок известна противнику. Разобьем ключ на две 40-битных половины и будем опробовать каждую из них, строя две таблицы:

$$z^{(k_1k_2k_3k_4)} = g_{k_1} g_{k_2} g_{k_3} g_{k_4}(x); \quad k_1, k_2, k_3, k_4 = 0,...,M\text{-}1,$$

$$\tilde{z}^{(k_5k_6k_7k_8)} = g_{k_8}^{-1} g_{k_7}^{-1} g_{k_6}^{-1} g_{k_5}^{-1}(y); \quad k_5, k_6, k_7, k_8 = 0,...,M\text{-}1.$$

Эти вычисления потребуют $2(1024)^4 = 2^{41} \approx 2 \cdot 10^{12}$ операций опробования и памяти $\approx 2^{41} \times l$ байт.

Проведем единое упорядочивание двух таблиц и выбор пар одинаковых последовательностей *z*. Сложность упорядочивания оценивается следующим образом

$$2^{41} \cdot l \cdot \ln(2^{41} l) \approx 41 \cdot 2^{41} \cdot l \cdot \ln 2 \approx 8 \cdot l \cdot 10^{13}$$

операций.

2.4. Метод “разделяй и побеждай”.

Пусть по-прежнему даны множества *X* открытых текстов, *Y* - шифрованных текстов, *K* - ключей, и шифр $T: X \times K \to Y$..

Предположим, что множество ключей *K* допускает представление $K = K_1 \times K_2$. То есть каждый ключ $k \in K$ может быть записан в виде вектора $k = (k_1, k_2)$, где $k_1 \in K_1$, $k_2 \in K_2$.

Предположим дополнительно, что существует критерий *h,* позволяющий при известных *x* и *y* проверять правильность компоненты k_1 в ключе $k = (k_1, k_2)$. То есть для любого k_2 функция $h(x,y,k_1) = 1$, если $\exists k_2 \in K_2$ такой, что $T(x,(k_1,k_2)) = y$, и $h(x,y,k_1) = 0$, если $\forall k_2 \in K_2$ $T(x,(k_1,k_2)) \neq y$.

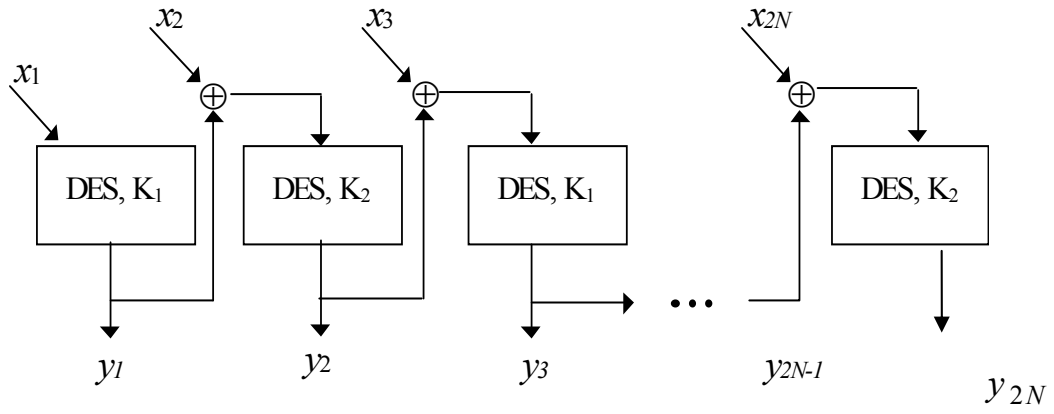
Если известные открытый и шифрованный тексты достаточно длинные, то достаточно часто существует единственный ключ *k* такой, что $T(x, k) = y$, поэтому сделанное предположение допустимо. Тогда метод “разделяй и побеждай” предлагает опробовать сначала элементы множества K_1, отбраковывая варианты k_1 по критерию *h.* Отсюда мы определим k_1 - первую компоненту вектора $k = (k_1, k_2)$. Затем, используя уравнение $T(x,(k_1,k_2)) = y$ относительно неизвестного параметра $k_2 \in K_2$, опробуем варианты k_2. Трудоемкость определения компоненты k_1 равна в среднем

$\frac{|K_1|}{2}$ операций опробования, соответственно трудоемкость определения k_2 равна в среднем $\frac{|K_2|}{2}$ операций опробования. Тогда в среднем трудоемкость метода равна $\frac{|K_1|}{2}+\frac{|K_2|}{2}$ против $\frac{|K_1|\cdot|K_2|}{2}$ при методе полного перебора.

Данный метод является обобщением изложенного в п. 2.2 аналитического метода подбора системы уравнений, зависящих от части ключей.

Основная проблема при применении метода “разделяй и побеждай” [Gol. 95] состоит в нахождении критерия *h,* который зависит от конкретного преобразования *T* или вообще может не существовать. Иногда критерий *h* может описываться вероятностно-статистической моделью. Мы рассмотрим такой пример в разделе “Статистические методы анализа”.

Пример. Рассмотрим вариант двойного DES. Пусть используются два ключа DES k_1 и k_2. Открытый текст *x* разбивается на последовательность 64-битных блоков $x = x_1, x_2, \ldots, x_{2N}$, а последовательность 64-битных блоков шифрованного текста $y = y_1, y_2, \ldots, y_{2N}$ получается поочередным применением алгоритма DES с ключами k_1 и k_2 к блокам открытого текста, сложенным с предыдущим блоком шифртекста. Схема шифрования приведена на следующем рисунке.



Пусть известны открытый и шифрованный тексты *x, y* и *N* доста

точно велико. Для применения метода “разделяй и побеждай” выделим четные и нечетные пары блоков в (x,y): $A=\{(x_{2k+1},y_{2k+1}),k=0,N-1\}$ и $B=\{(x_{2k},y_{2k}),k=1,...,N\}$. Преобразуем эти множества в следующие:

$$A'=\{(x_{2k+1}\oplus y_{2k},y_{2k+1}),k=0,...,N-1,y_0=0\},$$

$$B'=\{(x_{2k}\oplus y_{2k-1},y_{2k}),k=\overline{1,N}\}.$$

Блоки из A' получены по ключу k_1. Определим его опробованием, затратив в среднем $\frac{2^{56}}{2}$ операций опробования. В качестве *h* здесь берем функцию равную единице, когда $\mathrm{DES}(x_{2k+1}\oplus y_{2k},k_1)=y_{2k+1}$, $k=\overline{0,N-1}$. Затем ищем k_2, опираясь на множество B' и уравнение $\mathrm{DES}(x_{2k}\oplus y_{2k-1},k_2)=y_{2k}$. Определение k_2 потребует в среднем $\frac{2^{56}}{2}$ операций опробования. Тогда трудоемкость определения ключа $k=(k_1,k_2)$ в среднем равна 2^{56} операций опробования.

2.5. Методы криптоанализа при неравновероятной гамме. Расстояние единственности.

Пусть открытые тексты - слова длины *n* в алфавите *Z/m*. Для того, чтобы объяснить математическую теорию, разработанную для решения задачи, предположим, что гамма может принимать только *r* значений. Множество открытых X_n и шифрованных текстов Y_n погружены в пространство V_n всех последовательностей длины *n* в алфавите *Z/m*. Следуя [Ш., стр. 375], операцию дешифрования можно представить графически в виде ряда линий, идущих от каждого шифртекста $y\in Y_n$ к различным $x\in V_n$. В сделанных выше предположениях каждый $y\in Y_n$ связан ровно с $k=r^n$ различными $x\in V_n$. Обозначим их $x^{(1)},...,x^{(k)}$. Среди линий, идущих от y к $x^{(1)},...,x^{(k)}$ имеется открытый текст x_0. Если мы знаем признаки открытого текста (например, читаемость) и среди $x^{(1)},...,x^{(k)}$ ровно один текст x_0 удовлетворяет этим признакам, то тем самым процедура дешифрования завершена. Однако возможна ситуация, когда несколько текстов среди $x^{(1)},...,x^{(k)}$ удовлетворяют признакам открытого текста. Тогда цель дешифрования - получение достоверной информации из раскрытого открытого текста - недостигнута. Ясно, что на возможный исход дешифрования влияет *r*. Если *r* = 1, то от *y* ведет всего одна линия к *x* и, по условию, этот текст и есть открытый. Проблемы неоднозначности здесь нет. Если *r*

= m, то от y ведет m^n различных линий ко всем элементам V_n. Значит, любой текст из X_n является возможным открытым текстом, что не дает нам возможности достигнуть цели дешифрования и этот случай обязательно порождает неоднозначность прочтения криптограммы. Для того, чтобы понять при каких r возможно однозначное дешифрование, а при каких r невозможно, рассмотрим следующую модель. На множестве ключей задана равномерная мера, т.е. любая допустимая гамма появляется с вероятностью $1/|K|$. Мы также предположим, что для любого заданного шифртекста y, полученного из открытого текста x_0, все линии, связывающие y с $x^{(1)},...,x^{(k)}$, кроме (y, x_0), получены случайным и равновероятным выбором с возвращением из (m^n - 1) возможностей. Это предположение было впервые введено Шенноном [Ш.Б стр.374] при определении “случайного шифра” (фактически мы рассматриваем частный случай “случайного шифра”). Если модель с этим и следующими предположениями позволит оценивать r, допускающее однозначное дешифрование, и полученная оценка может быть подтверждена экспериментами, то теория правильно отражает суть вопроса.

Обозначим ξ_i случайную величину, равную 1, если $x^{(i)}$ является открытым текстом, и 0 в противном случае. Тогда число открытых текстов η без x_0 среди $x^{(1)},...,x^{(k)}$ равно

$$\eta = \sum_{i=1}^{k} \xi_i \text{ -1.}$$

Тогда $E\eta = \sum_{i=1}^{k} E\xi_i$ -1. Для всех $x^{(i)}$, которые не равны x_0, $E\xi_i = \frac{|X|}{m^n - 1}$. В случае $x^{(i)} = x_0$, $E\xi i = 1$. Тогда

$$E\eta = (k\text{-}1)\ \frac{|X|}{m^n - 1}\ . \qquad (1)$$

Если соотношение параметров таково, что $E\eta << 1$, то по оценке Маркова

$$P(\eta \le 1) \ge E\eta << 1.$$

Это означает, что появление ложных открытых текстов маловероятно или что x_0 выделяется однозначно с большой вероятностью. Для изучения соотношений между параметрами необходимо оценить $|X|$.

До сих пор мы представляли множество открытых сообщений x как подмножество множества V_n всех слов длины n. Однако нет оснований

считать, что какая-либо последовательность не может стать сообщением. Просто вероятность встретить в качестве сообщения одни слова много больше, чем другие. Поэтому предполагают, что все последовательности длины n упорядочены в соответствии с их вероятностями появления в общении на данном языке. Тогда собственно осмысленные (в бытовом понимании этого слова) выражения относятся к более вероятным, чем последовательности, смысл которых нам неясен (трудно представить себе общение при помощи труднопроизносимых буквосочетаний).

Поскольку значения k, m^n и $|X|$ велики, то оценку $|X|$ можно сделать асимптотически, предполагая n большим. Такие оценки Шеннон разработал в работе “Математическая теория связи” [Ш., с. 265], к которой сейчас и обратимся.

Пусть

$$p(a_1),...,p(a_m) \qquad (2)$$

вероятности появления букв на фиксированном месте i в открытом сообщении длины n ($1 \le i \le n$). Предположим, что буквы в сообщении появляются независимо друг от друга с одним и тем же распределением (2). Подобная модель открытых сообщений является грубой, но, как и выше, оценка для r, близкая к экспериментальной, покажет, что наш механизм анализа отражает сущность проблемы. Обозначим через ν_i , $i = 1,...,m$, частоты букв $a_1,..., a_m$ в последовательности x. Тогда вероятность выбора x в нашей схеме равна

$$P(x) = p^{\text{Ч}}(a_1)\cdots p^{\nu_m}(a_m).$$

Будем считать, что $p(a_i) > 0$, $H = -\sum_{i=1}^{m} p(a_i) \log_2 p(a_i)$. Теперь мы можем сформулировать следующую теорему Шеннона.

Теорема 1 (К. Шеннон). *Для любых $\varepsilon > 0$ и $\delta > 0$ можно найти такое n_0, что для любого $n > n_0$ последовательности из V_n распадаются на два непересекающихся класса B и $\overline{B}$ так, что*

1) $P(\overline{B}) < \varepsilon$

2) $\forall x \in B$ выполняется неравенство

$$\left| \frac{\log_2 P^{-1}(x)}{n} - \text{H} \right| < \delta.$$

Доказательство. Возьмем произвольные малые $\varepsilon > 0$ и $\delta > 0$ и рассмотрим события

$$\overline{B}_i = \{x \in V, \ | \nu_i - np(a_i) | > \delta n \}, \ i = 1,...,m.$$

По закону больших чисел $\exists\ n_0^{(i)}$, что $\forall\ n > n_0^{(i)}$

$$P(\overline{B}_i) < \frac{\varepsilon}{m}. \qquad (3)$$

Определим $\overline{B} = \bigcup_{i=1}^{m} \overline{B}_i$. Тогда из (3) для $n > \max_i \{n_0^{(i)}\}$

$$P(\overline{B}) \le \sum_{i=1}^{m} P(\overline{B}_i) < \varepsilon. \qquad (4)$$

Множество $B = \bigcap_{i=1}^{m} B_i$ может быть представлено в следующем виде:

$$B = \{x \in V, \forall\ i\ (i=1,..., m) \mid \nu_i - np(a_i) \mid \le \delta n\}. \qquad (5)$$

Обозначим $\alpha_i = \nu_i - np(a_i)$, $i=1,..., m$. Из (5) следует, что $\forall\ i\ \ -\delta n \le \alpha_i \le \delta n$.

Выразим $P(x)$, $x \in B$, через введенные параметры.

$$P(x) = (p(a_1))^{np(a_1)+\alpha_1} \cdot\cdot\cdot\cdot (p(a_m))^{np(a_m)+\alpha_m}.$$

Тогда

$$\log_2 \frac{1}{P(x)} = -n \sum_{i=1}^{m} p(a_i) \log_2 p(a_i) - \sum_{i=1}^{m} \alpha_i \log_2 P(a_i).$$

Получаем, что $\forall\ x \in B$

$$\left| \frac{log_2 P^{-1}(x)}{n} - H \right| \le \frac{1}{n} \sum_{i=1}^{m} |\alpha_i|\,|log_2 P(a_i)| \le \delta \sum_{i=1}^{m} |log_2 p(a_i)|.$$

Поскольку $p(a_i) > 0$, то $\sum_{i=1}^{m} |log_2 p(a_i)| = \text{const}$, не зависит от n. Теорема доказана.

Пусть $0 < \varepsilon < 1$, - произвольное малое число. Пусть все последовательности длины n расположены в порядке убывания вероятностей их появления. Как отмечалось выше, множество открытых сообщений моделируется начальным участком таких последовательностей. Обозначим через $\beta_n(\varepsilon)$ - число наиболее вероятных последовательностей таких, что сумма их вероятностей $\ge 1-\varepsilon$, а сумма вероятностей любого набора из $(\beta_n(\varepsilon) - 1)$ этих последовательностей $< 1-\varepsilon$. Следующая теорема показывает, что при $n \to \infty$ множество последовательностей, составляющих в нашей модели открытый текст, не зависит от ε.

Теорема 2 (К.Шеннон). *Для любого* $\varepsilon > 0$

$$\lim_{n\to\infty} \frac{\log_2 \beta_n(\varepsilon)}{n} = H.$$

Д о к з а т е л ь с т в о. Пусть $Q_n(\varepsilon)$ - множество наиболее вероятных последовательностей таких, что P($Q_n(\varepsilon)$)≥ 1-ε, а исключение любой последовательности из $Q_n(\varepsilon)$ образует множество, вероятность которого меньше 1-ε. Возьмем малое δ>0 и рассмотрим множество B из предыдущей теоремы. Тогда для $x \in B$

$$H - \delta < \frac{-\log_2 P(x)}{n} < H + \delta$$

или

$$- n(H-\delta) > \log_2 P(x) > - n(H+\delta).$$

Отсюда

$$2^{-n(H+\delta)} < P(x) < 2^{-n(H-\delta)}. \qquad (6)$$

Рассмотрим оценки для $Q_n(\varepsilon)$. По определению $Q_n(\varepsilon)$ $P(Q_n(\varepsilon))$≥ 1-ε. Поскольку

$$Q_n(\varepsilon) = (Q_n(\varepsilon) \bigcap \overline{B}) \cup (Q_n(\varepsilon) \bigcap B),$$

тогда

$$P(Q_n(\varepsilon)) = \sum_{x\in Q_n(x)} P(x) = \sum_{x\in Q_n(x)\bigcap B} P(x) + \sum_{x\in Q_n(x)\bigcap \overline{B}} P(x).$$

Для второй суммы справедлива оценка

$$P(Q_n(\varepsilon) \bigcap \overline{B}) \le P(\overline{B}) < \varepsilon.$$

$$P(Q_n(\varepsilon) \bigcap B)) = \sum_{x\in Q_n(x)\bigcap B} P(x) < \sum_{x\in Q_n(x)\bigcap B} 2^{-n(H-\delta)} =$$

$$= | Q_n(\varepsilon) \bigcap B) | \cdot 2^{-n(H-\delta)} \le | Q_n(\varepsilon) | \cdot 2^{-n(H-\delta)} =$$

$$= \beta_n(\varepsilon) 2^{-n(H-\delta)}.$$

Следовательно,

$$1-\varepsilon \le P(Q_n(\varepsilon)) < \beta_n(\varepsilon) 2^{-n(H-\delta)} + \varepsilon.$$

Отсюда получаем

$$\beta_n(\varepsilon) > 2^{n(H-\delta)} (1- 2\varepsilon).$$

Докажем, что множество $Q_n(\varepsilon)$ содержит минимальное число последовательностей среди всех множеств C, таких что $P(C) \ge 1-\varepsilon$. Доказательство будем вести от противного. $\forall x \in Q_n(\varepsilon)$, $x \notin C$ и $\forall y \in C$, $y \notin Q_n(\varepsilon)$, такие последовательности, что $P(x) \ge P(y)$, так как в $Q_n(\varepsilon)$ содержатся наиболее вероятные последовательности. Предположим, что

$$| C \setminus Q_n(\varepsilon) | \cdot < | Q_n(\varepsilon) \setminus C |$$

(т.е. $Q_n(\varepsilon)$ - не минимальное множество). Тогда

$$\sum_{x\in C\setminus Q_n(\varepsilon)} P(x) < \sum_{x\in Q_n(\varepsilon)\setminus C} P(x),$$

так как справа слагаемых хотя бы на 1 больше и все они не меньше слагаемых в левой сумме. Пусть y_0 имеет наименьшую вероятность среди всех $y \in Q_n(\varepsilon)\setminus C$. Следовательно,

$$\sum_{x\in C\setminus Q_n(\varepsilon)} P(x) \le \sum_{x\in (Q_n(\varepsilon)\setminus C)\setminus\{y_0\}} P(x).$$

Поскольку

$$Q_n(\varepsilon)\setminus\{y_0\} = (Q_n(\varepsilon)\cap C)\cup((Q_n(\varepsilon)\setminus C)\setminus\{y_0\}),$$

то, учитывая, что по определению множества $Q_n(\varepsilon)$

$$P(Q_n(\varepsilon)\setminus\{y_0\}) < 1 - \varepsilon,$$

получим

$$P(C) = P(Q_n(\varepsilon)\cap C) + P(C\setminus Q_n(\varepsilon)) \le P(Q_n(\varepsilon)\cap C) +$$
$$+ P(Q_n(\varepsilon)\setminus C\setminus\{y_0\}) = P(Q_n(\varepsilon)\setminus\{y_0\}).$$

Тогда

$$P(C) < 1 - \varepsilon,$$

что противоречит предположению. Следовательно, $Q_n(\varepsilon)$ - минимальное множество векторов, имеющих максимальную вероятность. Отсюда и из (6)

$$|Q_n(\varepsilon)| \le |B| = \sum_{x\in B} 1 < \sum_{x\in B} \frac{p(x)}{2^{-n(H+\delta)}} \le 2^{n(H+\delta)}.$$

Таким образом,

$$2^{n(H-\delta)} < |Q_n(\varepsilon)| < 2^{n(H+\delta)}$$

или

$$H-\delta < \frac{\log_2 \beta_n(\varepsilon)}{n} < H+\delta.$$

Следовательно,

$$\lim_{n\to\infty} \frac{\log_2 \beta_n(\varepsilon)}{n} = H,$$

что и требовалось доказать.

Множество открытых текстов можно представить как объединение B и $\overline{B}$. Множество $\overline{B}$ имеет маленькую вероятность, множество B оценивается по теореме 2.

$$|X| \approx 2^{nH(x)}.$$

Следовательно,

$$E\eta = (r^n\text{-}1)\ \frac{|X|}{m^n - 1} \to 0$$

при

$$E\eta = (\frac{r}{m} \cdot 2^H)^n \to 0.$$

Это выполняется, когда

$$\frac{r}{m} \cdot 2^H < 1,$$

то есть при r, удовлетворяющих неравенству

$$log_2 r - log_2 m + H < 0, \qquad (7)$$

возможно однозначное дешифрование.

2.6. Перекрытия гаммы.

Рассмотрим шифр гаммирования $y_i = x_i + \gamma_i$, i = 1,2,...,*n*, где $x = x_1 x_2 .....$- открытый текст, $y = y_1 y_2 ....$ - шифрованный текст и $\gamma = \gamma_1 \gamma_2 .....$ - ключ шифра гаммирования по mod *m*.

Предположим, что одна γ использовалась дважды для зашифрования двух открытых текстов *x* и *x*'. Дешифровщик получил два шифртекста *y* и *y*', полученных при зашифровании открытых текстов *x* и *x*' одной γ. Тогда

$$y - y' = x + \gamma - x' - \gamma' = x - x'.$$

Таким образом, дешифровщик имеет разность открытых текстов $x - x'$. Идея следующего метода изложена в [Ш., с.396]. Пусть дешифровщик угадал, что в *x* начиная с места *i* стоит слово $a_1 a_2 ... a_{r+1}$. Тогда

$$y'_i - y_i + a_1 = x'_i$$

.....

$$y'_{i+r} - y_{i+r} + a_{r+1} = x'_{i+r}.$$

То есть, дешифровщик может сразу прочитать, что написано в телеграмме начиная с места *i* по место *i+r* в открытом тексте *x'*. Если дешифровщик лишь предполагает, что начиная с места *i* в *x* стоит слово $a_1 a_2 ... a_{r+1}$, то читаемость участка $x'_i ... x'_{r+1}$ в открытом тексте *x'* является критерием правильности его предположения. Если дешифровщик угадал слово в *x* и подтвердил его правильностью в *x*', то он может прогнозировать текст в *x* и в *x*' слева и справа от *i* и *i+r*. Правильность догадки тут же подтверждается читаемостью второго текста. Таким образом, можно прочитать оба текста. Этот метод дешифрования называется "протяжкой вероятного слова".

Если есть две криптограммы *y* и *y*' (или два куска криптограмм), то

возникает вопрос о том, можно ли узнать до протяжки вероятного слова о том, зашифрованы ли они одной γ. Для решения данной задачи опять рассмотрим простейшую модель открытого текста - последовательность независимых испытаний по полиномиальной схеме с вероятностями $p(a_1),..., p(a_m)$ (распределение P). Тогда случайная величина $\xi_i = x_i - x_i'$ имеет распределение $P^* = P*P$, где P^* - свертка распределения P с собой. Если P - неравновероятное распределение, то $P^*=(p^*(a_1),...,p^*(a_m))$ также неравновероятное распределение. Тогда $y - y' = x - x'$- независимые случайные величины с распределением P^*. Если x зашифровывается с помощью γ, а x' - с помощью γ', то

$$y - y' = x - x' + \gamma - \gamma'.$$

В наших предположениях $\gamma - \gamma'$ - равновероятно распределенные случайные величины. Следовательно, $y - y'$ - также независимые и равновероятно распределенные случайные величины. Значит, статистический критерий, проверяющий гипотезу H_0 о равновероятности $y - y'$ против альтернативы H_1, что $y - y'$ имеет распределение P^*, дает ответ о наличии перекрытия в y и y'.

Обычно схема получения гаммы следующая: имеется конечный автомат A без входа, в котором начальное состояние есть ключ k. Функция выхода этого автомата есть γ для шифрования. Например, линейный регистр сдвига с линейной и нелинейной обратной связью. При таком способе получения γ повтор γ получается, когда автомат начинает вырабатывать периодическую последовательность. Период последовательности возникает обязательно из-за конечности множества состояний автомата. Если период небольшой, то его можно опробовать и, применяя критерий на перекрытие, найти, а затем дешифровать криптограммы. Если n - число состояний автомата велико, то можно получить большой период, а, следовательно, мало шансов на перекрытие.

Пусть S- множество состояний автомата A, k - случайное начальное состояние. Период возникает, когда возникает повторение в последовательности $A(k)$, $A^2(k)$,.... Пусть A - случайная равновероятная подстановка на $\{1,\ldots,n\}$. Тогда возврат возможен только в точку k. Если ι - длина полученного цикла, и случайная величина $\xi_i = 1$, если длина цикла равна i, и $\xi_i = 0$ в противном случае, то

$$P(\xi_i = 1) = \frac{C_{n-1}^{i-1}(i-1)!(n-i)!}{n!} = \frac{(n-1)!(n-i)!(i-1)!}{(i-1)!(n-i)!n!} = \frac{1}{n}.$$

Тогда

$$\iota = \sum_{i=1}^{n} i\xi_i \,.$$

Отсюда

$$E\iota = \sum_{i=1}^{n} iE\xi_i = \sum_{i=1}^{n} \frac{i}{n} = \frac{n+1}{2}.$$

При $n = 2^{64}$ $E\iota \approx 10^{18}$, что дает мало шансов ожидать перекрытия даже при очень большой интенсивности переписки.

Если A - случайное отображение (не взаимно-однозначное) и η - длина цикла, а h - длина подхода, то тогда $h \sim \sqrt{n}$, $\eta \sim \sqrt{n}$ и $h+\eta \sim \sqrt{n}$. Следовательно, при $n = 2^{64}$ $h+\eta \sim 2^{32} \sim 10^{9}$, что является не очень большой величиной и можно ожидать перекрытия гаммы.

2.7. Корреляционные атаки на поточные шифры.

Чаще всего цель статистических методов криптоанализа [Meier 89] состоит в том, чтобы построить критерий *H,* необходимый для применения метода “разделяй и побеждай”. Оказывается, что большинство поточных шифров дают возможность применять против них корреляционные методы атак. Идея метода состоит в том, чтобы заменить исходное преобразование в шифре другим, зависящим от части ключа, что открывает возможность применения метода “разделяй и побеждай” или аналитических методов. В последнем случае часто стремятся свести задачу криптоанализа к решению системы линейных уравнений. Замена одного преобразования другим осуществляется так, чтобы сохранялась корреляционная связь с известной последовательностью знаков шифра, полученной при настоящем преобразовании. Тогда при опробовании части ключа эта статистическая связь дает критерий *H* для определения правильного варианта опробоваемой части ключа. Если происходит сведение к линейной системе уравнений, то указанная статистическая связь позволяет построить линейную систему относительно элементов ключа.

Пример 1. Рассмотрим генератор поточного шифра [Meier 89], в котором имеется известная двучленная линейная рекуррентная последовательность, усложненная некоторой булевой функцией f (неравновероятной), как это представлено на рис. 1.

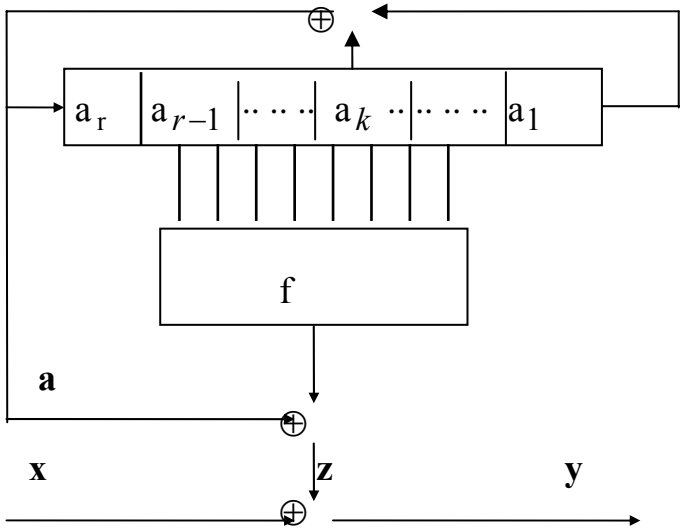


Рис.1

$$a_{i+r} = a_i + a_{i+k-1}\ ,\ 1 \le k \le r,\ i = 1, 2, \ldots. \qquad (1)$$

Мы считаем, что известна последовательность $\boldsymbol{z} = (z_1,...,z_N)$ длины N. Таким образом, задача будет решаться в предположении, что известны открытые и шифрованные тексты. Выходная последовательность линейного регистра будет обозначаться $\boldsymbol{a} = (a_1,\ldots,a_N)$. Неизвестным ключом является начальное заполнение регистра, которое выбирается случайно и равновероятно.

Для целей криптоанализа примем следующую модель схемы 1.

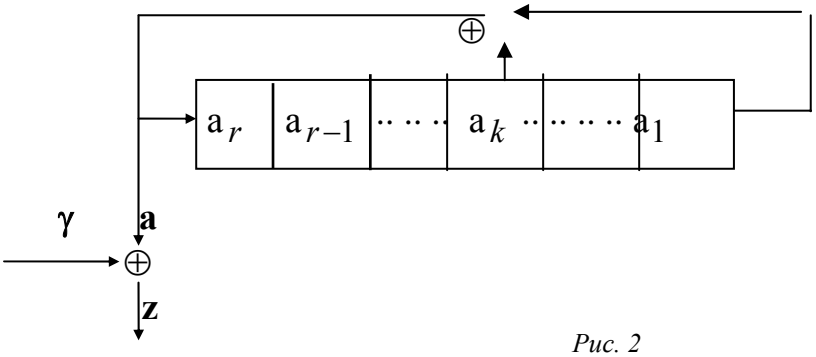


Рис. 2

Здесь $\gamma = (\gamma_1,...,\gamma_N)$ - последовательность независимых одинаково распределенных {0, 1} случайных величин с вероятностью 0, равной p. Соответствие вероятностной модели на рис. 2 детерминированному автомату рис. 1 может быть оправдана, если нам удастся найти начальное заполнение регистра. Поэтому мы и далее будем строить грубые вероятностные модели, оправдывая сделанные допущения достижимостью конечной цели. Если ключ определить не удается, то это может произойти из-за ошибки стати-

стического критерия, или неадекватности модели. В любом случае криптоанализ надо начинать заново.

Отметим, что величина a_n для каждого n присутствует в нескольких линейных соотношениях. Например,

$$a_n + a_{n-r} + a_{n-r+k-1} = 0, \quad (2)$$

$$a_{n+r-k+1} + a_{n-k+1} + a_n = 0, \quad (3)$$

$$a_{n+r} + a_n + a_{n+k-1} = 0, \quad (4)$$

Характеристический многочлен рекурренты равен

$C(x) = 1 + x^{k-1} + x^r$.

Тогда справедливо следующее равенство:

$\forall\ i$ при $j = 2^i$ $(C(x))^j = C(x^j)$.

Отсюда мы можем получить новые трехчленные соотношения.

Возможны другие производные соотношения, если в (2) вместо a_{n-r} подставить правую часть равенства:

$a_{n-r} = a_{n-2r} + a_{n-2r+k-1}$.

Или в (2) вместо $a_{n-r+k-1}$ подставить правую часть равенства

$a_{n-r+k-1} = a_{n-2r+k-1} + a_{n-2r+2k-2}$,

и т.д. Таким образом, можно набрать m линейных соотношений относительно a_n.

Согласно модели для любого n $p = P(z_n = a_n) > 0{,}5$. Фиксируем a_n и индекс n будем далее опускать. Для m линейных соотношений из рекурренты, обозначив через b_i, i=1,...,m, суммы слагаемых без a_n, получим следующую систему

$$\begin{aligned} &a + b_1 = 0 \\ &a + b_2 = 0 \\ &\text{.....} \\ &a + b_m = 0 \end{aligned} \quad (5)$$

Реально вместо a будем подставлять z_n. Обозначим значение сумм z_i с линейными формами на соответствующих местах в формуле (5) через y_i. Получим линейные формы

$$\begin{aligned} &z + y_1 = L_1 \\ &z + y_2 = L_2 \\ &\text{.....} \\ &z + y_m = L_m \end{aligned} \quad (6)$$

Если $z = a$, $y_i = b_i$, то

$L_i = 0$. i= 1,...,m. (7)

При достаточно большом количестве m соотношений в (5) мы надеемся, что удастся статистически различить случаи $z = a$ от $z \neq a$. Если статистически эти случаи различаются, то среди большого числа z_n найдутся такие, для которых такое статистическое обоснование $z = a$ будет хорошо заметно. Тогда выразим все a_n, соответствующие таким случаям ($z = a$) через начальное заполнение регистра сдвига. Мы получим систему линейных уравнений, у которых правые части с большой вероятностью совпадают со значениями a_n. Если система решается, то получим ключ. Если система не решается, то надо опробовать малое число уравнений с неправильно определенной правой частью. Решая такие системы, находим ключ.

2.8. Статистические модели.

Для того, чтобы оценить реализуемость метода, рассмотрим следующую вероятностную модель. Рассмотрим набор случайных величин

$A = \{a, b_{ij}, i=1,...,m, j=1,...,t\}$,

удовлетворяющих системе уравнений:

$$a + \sum_{j=1}^{t} b_{ij} = 0,\ i=1,...m.$$

Замечание 1. В модели, рассмотренной в п. 2.6, t=2.

Аналогично рассмотрим набор случайных величин

$Z = \{z, y_{ij}, i=1,...,m, j=1,...,t\}$.

Эти случайные величины представляют z_n. Два набора связаны следующим соотношениями

$P(z=a) = p, P(b_{ij} = y_{ij}) = p.$

Замечание 2**.** Такие соотношения можно получить следующим образом. Так, например, в модели (2)

$z = a + \gamma,$

$y_{ij} = b_{ij} + \gamma_{ij},$

где γ - независимые, одинаково распределенные случайные величины с $P(\gamma = 0) = p.$

Обозначим

$$b_i = \sum_{j=1}^{t} b_{ij},\ i=1,...m,$$

$$y_i = \sum_{j=1}^{t} y_{ij},\ i=1,...m.$$

Положим

$L_i = z + y_i$, i=1,...m.

Пусть вероятность $s(t,p) = s = P(y_i = b_i)$ не зависит от i. По формуле полной вероятности получим рекурренту

$s(t,p) = ps(t\text{-}1,p) + (1\text{-}p)(1\text{-}s(t\text{-}1,p))$,

$s(1,p) = p$.

Замечание 3. Для t=2 $s(2,p) = p^2 + (1\text{-}p)^2$.

Обозначим через B_k события, состоящие в том, что k из m линейных форм L_i равны 0. Тогда следующий вывод "$z = a$" определяется апостериорной вероятностью

$$P(z=a|\, B_k) = \frac{\binom{m}{k} ps^k (1-s)^{m-k}}{\binom{m}{k} ps^k (1-s)^{m-k} + \binom{m}{k} (1-p)(1-s)^k s^{m-k}} = p^*.$$

Это апостериорная вероятность того, что z=a. Аналогично

$$P(z\neq a|\, B_k) = \frac{\binom{m}{k} (1-p)(1-s)^k s^{m-k}}{\binom{m}{k} ps^k (1-s)^{m-k} + \binom{m}{k} (1-p)(1-s)^k s^{m-k}} = 1\text{-}p^*.$$

Идея состоит в том, что мы интуитивно ожидаем, что p^* увеличивается по сравнению с p, если $z = a$ и уменьшается, если $z \neq a$. Поэтому найдем математическое ожидание p^* в двух случаях $z = a$ и $z \neq a$. При $z = a$

$$E_0(p^*) = E(p^* | z = a) =$$

$$= \sum_{k=0}^{m} \binom{m}{k} \frac{ps^k (1-s)^{m-k}}{ps^k (1-s)^{m-k} + (1-p)(1-s)^k s^{m-k}} s^k (1-s)^{m-k}.$$

Далее при $z \neq a$

$$E_1(p^*) = E(p^* | z \neq a) =$$

$$= \sum_{k=0}^{m} \binom{m}{k} \frac{p(1-s)^{m-k} s^k}{ps^k (1-s)^{m-k} + (1-p)(1-s)^k s^{m-k}} s^{m-k} (1-s)^k.$$

При p= 3/4, t=2, m=20 получим

$E_0(p^*) = 0{,}9$,

$E_1(p^*) = 0{,}3$.

Осталось оценить число допустимых соотношений m как функции от длины регистра r и длины текста N. Пусть t-членное соотношение получено с использованием равенства

$(C(x))^j = C(x^j)$, j $= 2^i$, $\forall$ i= 1,...,m.

Тогда длина задействованного участка при i=m равна $r2^m$. Таких соотношений N- $r2^m$>0. Тогда общее число соотношений равно

$$T = \sum_{m=0}^{\log_2 \frac{N}{r}} (N - 2^m r) = N(log_2 \frac{N}{r} +1) - \sum_{m=0}^{\log_2 \frac{N}{r}} 2^m r =$$

$$= N(log_2 \frac{N}{r} + 1) - (2^{\log_2 \frac{N}{r}+1} -1)r =$$

$$= Nlog_2 \frac{N}{r} + N - (\frac{2N}{r} - 1)r =$$

$$= Nlog_2 \frac{N}{2r} + r - N. \qquad (1)$$

Каждое соотношение (1) связано с t+1 знаками последовательности z. Поэтому среднее число m соотношений на один знак равно

$$m = \frac{T(t+1)}{N} = (log_2 \frac{N}{2r} + \frac{r}{N} - 1)(t+1).$$

Для наших приложений $\frac{r}{N}(t + 1) << 1$. Отсюда из формулы (1) получим приближенное равенство

$$m \approx (t + 1)\ log_2 \frac{N}{2r}.$$

2.9. Линейный криптоанализ блочных шифров.

Линейный криптоанализ [Matsui 93] является атакой при известном открытом и шифрованном текстах на итеративные шифры типа DES, в которых “криптографически слабая” функция повторяется r раз. Предполагается, что открытый текст выбирается случайно и равновероятно, а подключи в каждом раунде независимы друг от друга.

Замечание 1. По-видимому, не имеет значения каким образом выбирается открытый текст.

Часто линейный криптоанализ является наиболее эффективным методом, тогда сложность линейного криптоанализа итеративных блочных шифров может являться мерой их слабости.

Линейный криптоанализ Мацуи *r*-раундового DES требует значительного числа открытых текстов. Это следует из следующей таблицы.

Количество раундов	Количество известных открытых текстов для нахождения ключа
8	2^{21}
12	2^{33}
16	$2^{47} \xrightarrow{Crypto'94} 2^{43}$

$\rightarrow$ *Eurocrypt*'93

Сложность атаки связана с количеством необходимых известных открытых текстов, т.к. для любой пары (открытый текст, шифртекст) требуется небольшое количество вычислений для реализации алгоритма.

Например, атака на *r*=8 DES занимает 40 сек. на рабочей станции, а атака на *r*=12 DES занимает 50 час, что примерно в 4500 раз больше.

Эта атака может быть проведена, если имеется только шифртекст. Мацуи были получены следующие результаты:

- если известно, что открытый текст на английском языке, представленный в ASCII кодах, то при *r*=8 DES вскрывается при 2^{29} известных шифртекстах;
- если открытый текст является случайным ASCII кодом, то при r=8 DES вскрывается при 2^{37} известных шифртекстах.

Рассмотрим основную идею линейного криптоанализа. Пусть S – {0,1} – случайная величина с $P\{S=1\}$=p. $\Delta(S) = |1-2p|$.

Лемма 1. Если $S_{(1)}, \ldots, S_{(n)}$ - независимые бинарные случайные величины, тогда

$$\Delta(S_{(1)} \oplus \ldots \oplus S_{(n)}) = \prod_{i=1}^{n} \Delta(S_{(i)}).$$

Доказательство. В силу независимости $S_{(1)}$ и $S_{(2)}$

$P(S_{(1)} \oplus S_{(2)} = 1)$ = $P(S_{(1)}=1)P(S_{(2)}=0)$ + $P(S_{(1)}=0)P(S_{(2)}=1)$ = $= p_1 + p_2 - 2p_1p_2$.

Тогда

$$\Delta(S_{(1)} \oplus S_{(2)}) = 1 - 2p(S_{(1)} \oplus S_{(2)} = 1) = 1 - 2p_1 - 2p_2 + 4p_1p_2 =$$

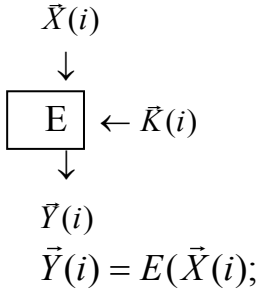
$= (1-2p_1)(1-2p_2).= \Delta(S_{(1)})\Delta(S_{(2)}).$

Утверждение леммы следует из-за ассоциативности операции $\oplus$:

$S_{(1)} \oplus (S_{(2)} \oplus S_{(3)}) = (S_{(1)} \oplus S_{(2)}) \oplus S_{(3)}$.

Заметим, что $0 \le \Delta(S) \le 1$. При этом $0 = \Delta(S)$ при $p = \frac{1}{2}$, а $\Delta(S) = 1$ при $P(S=1) = 1$ или $P(S=0) = 1$.

Рассмотрим схему произвольного итеративного блочного шифра в i-ом раунде.



$$\vec{Y}(i) = E(\vec{X}(i); \vec{K}(i)) \qquad (1)$$

Здесь E – функция шифрования, $\vec{X}(i)$ - блок открытого текста в i-ом раунде, $\vec{Y}(i)$ - блок шифртекста, $\vec{K}(i)$ - подключ, используемый в i-ом раунде. $\vec{Y}(i), \vec{X}(i) \in V_n$, $\vec{K}(i) \in V_m$, n – размер блока, m – размер подключа.

Обозначим через $(\vec{X}, \vec{\alpha}) = X_1\alpha_1 \oplus \ldots \oplus X_n\alpha_n = X_{i_1} \oplus \ldots \oplus X_{i_k} = X[i_1, \ldots, i_k]$- скалярное произведение двоичных векторов $\vec{X}$ и $\vec{\alpha}$, где $(\alpha_{i_1}, \ldots, \alpha_{i_k})$ единичные координаты вектора $\vec{\alpha}$.

Определение 1. Линейным статистическим аналогом нелинейной функции (1) будем называть случайную величину

$$S(i) = (\vec{Y}(i), \vec{\alpha}(i)) \oplus (\vec{X}(i), \vec{\beta}(i)) \oplus (\vec{K}(i), \vec{\gamma}(i)), \qquad (2)$$

если вероятность $P(S(i) = 1) = p \neq 1/2$ для случайно выбранного открытого текста $\vec{X}(i)$.

Отклонение $\Delta(S(i)) = |1-2p|$ определяет эффективность линейного статистического аналога (2).

Замечание 2. Массе [Massey 94] называет $S(i)$ тройной суммой и предлагает обобщение линейного криптоанализа, рассматривая тройную сумму для i-го раунда как случайную величину $S(i)$ вида

$S(i) = f_i(\vec{X}(i)) \oplus g_i(\vec{Y}(i)) \oplus h_i(\vec{K}(i))$,

где f_i ,g_i ,h_i - равновероятные булевы функции. В определении Мацуи f_i ,g_i ,h_i - линейные функции.

Определение 2. Последовательные тройные суммы $S(i+1)$ и $S(i)$ называются связанными, если $f_{i+1} = g_i$.

Из определения следует, что

$$S(i) \oplus S(i+1) = f_i(\vec{X}(i)) \oplus g_i(\vec{Y}(i)) \oplus h_i(\vec{K}(i)) \oplus f_{i+1}(\vec{X}(i+1))$$

$$\oplus g_{i+1}(\vec{Y}(i+1)) \oplus h_{i+1}(\vec{K}(i+1)),$$

$$\vec{Y}(i) = \vec{X}(i+1).$$

Определение 3. Если $S(1)$, …, $S(n)$ - связанные тройные суммы, то тогда

$$S_{1\dots n} = S(1) \oplus \dots \oplus S(n) = f_1(\vec{X}(1)) \oplus g_n(\vec{Y}(n)) \oplus \sum_{i=1}^{n} h_i(\vec{K}(i)) \qquad (3)$$

и $S_{1\dots n}$ называется n-раундовой тройной суммой.

В случае линейных функций формула (3) дает линейное соотношение

$$S_{1\dots n} = (\vec{X}(1),\ \vec{\alpha}) \oplus (\vec{Y}(n),\ \vec{\beta}) \oplus \sum_{i=1}^{n} (\vec{K}(i), \vec{\gamma}(i)), \qquad (4)$$

которое выполняется с вероятностью, определяемой $\Delta(S(1) \oplus \dots \oplus S(n))$ и леммой 1.

Определение 4. Эффективным линейным статистическим аналогом называется линейный статистический аналог (4) из заданного множества с наибольшим Δ.

Для применения линейного криптоанализа необходимо решить следующие задачи:

1. нахождения эффективного линейного статистического аналога и вычисление его вероятности;

2. определения ключа (или некоторых бит ключа) с использованием эффективного линейного статистического аналога.

Для нахождения эффективного линейного статистического аналога рассмотрим сначала задачу нахождения статистических аналогов для S-боксов алгоритма DES.

Нелинейная функция, реализующая a-ый S-бокс алгоритма DES, может быть записана в виде

$$\vec{Y} = F_a(\vec{X} \oplus \vec{K}), \quad a = \overline{1,8},$$

$$\vec{Y} \in V_4,\ \vec{X}, \vec{K} \in V_6. \qquad (5)$$

Пусть $1 \le i < 64$, $1 \le j < 16$, а $\vec{k}$ - двоичное разложение числа $k \in N$. Линейным статистическим аналогом каждого из уравнений (5) будет уравнение вида

$$(\vec{Y}, \vec{j}) = (\vec{X}', \vec{i}),$$

$$\vec{X}' = \vec{X} \oplus \vec{K}. \qquad (6)$$

Обозначим через $S_a(i, j),\ a = \overline{1,8},\ i = \overline{1,63},\ j = \overline{1,15}$, число ненулевых

$\vec{X} \in V_6$ для *a*-го *S*-бокса DES таких, что для *i* и *j* выполняется равенство (6). Когда $S_a(i, j) \neq 32$ можно сказать, что есть статистическая связь между входными и выходными битами *a*-го *S*-бокса.

Пусть

$$S_a^*(i^*, j^*) : |S_a^*(i^*, j^*) - 32| = \max |S_a(i, j) - 32| \qquad (7)$$

$$1 \le \mathrm{i} \le 63$$

$$1 \le \mathrm{j} \le 15$$

Тогда уравнение

$$(\vec{X}, \vec{i}^*) \oplus (\vec{Y}, \vec{j}^*) = (\vec{K}, \vec{i}^*) \qquad (8)$$

является эффективным линейным статистическим аналогом *a*-го *S*-бокса в классе всех линейных статистических аналогов вида (6) с вероятностью

$$p_a = \frac{S_a^*(i^*, j^*)}{64}, \qquad a = \overline{1,8}. \qquad (9)$$

Все значения для $S_5(i, j)$ приведены в таблице 1. Значение $S_5(i, j)$, имеющее наибольшее отклонение от $\frac{1}{2}$, помечено * в таблице 1. Следовательно, $S_5^*(i^*, j^*) = 12$, где $\vec{j}^* = (0,1,0,0,0,0)$, $\vec{i}^* = (1,1,1,1)$.

Отсюда эффективным линейным статистическим аналогом 5-го *S*-бокса является уравнение

$$\vec{Y}[1,2,3,4] \oplus \vec{X}[2] = \vec{K}[2], \qquad (10)$$

и это уравнение выполняется с вероятностью $p_5 = \frac{3}{16}$.

i	Значения *j*														
S(i, j)	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
1	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
2	36	30	34	30	34	28	32	36	32	34	30	34	30	32	28
3	32	30	38	30	30	36	28	32	32	30	38	30	30	36	28
4	34	30	32	32	34	30	32	32	34	34	36	28	30	30	32
5	34	34	28	32	42	26	28	32	34	22	32	36	30	34	36
6	30	28	26	30	28	34	32	32	30	32	30	26	24	34	32
7	34	32	34	30	40	38	32	28	38	32	26	30	32	26	28
8	32	34	38	32	32	30	26	30	34	36	20	34	38	28	36
9	28	38	30	32	28	26	26	38	30	32	28	34	26	24	28
10	36	32	32	30	26	34	34	34	34	30	34	36	28	28	32
11	36	36	36	38	34	30	30	30	30	30	34	32	24	28	32
12	34	32	30	32	34	36	42	30	36	30	24	30	36	26	28
13	38	32	34	32	30	36	22	30	32	30	36	30	40	26	32
14	30	30	32	30	36	32	34	30	32	36	34	28	38	30	28
15	30	30	40	38	36	32	34	34	36	40	30	40	26	34	32

16	34	30	32	32	30	26	24	32	30	30	28	32	34	42	12*
17	34	30	32	36	34	30	28	36	34	34	32	24	26	34	36
18	30	32	30	34	28	30	24	36	38	36	38	30	36	26	32
19	26	32	34	30	36	34	32	36	26	36	34	26	36	30	32
20	36	28	32	32	32	32	32	28	28	36	36	32	36	28	32
21	36	32	28	28	36	24	24	32	32	28	36	40	36	32	36
22	32	38	38	34	30	36	32	36	32	38	34	34	34	32	32
23	36	26	30	38	30	28	36	36	28	26	34	30	34	32	36
24	38	32	34	36	22	28	34	34	32	30	32	34	36	30	28
25	34	36	26	32	30	36	30	38	40	38	36	42	32	34	28
26	34	34	24	30	36	32	34	30	32	36	34	32	30	30	32
27	34	38	28	26	32	32	34	38	40	32	30	28	26	30	32
28	32	30	34	36	32	26	34	30	38	28	32	34	30	32	32
29	36	30	38	24	32	30	34	42	30	24	24	34	34	32	36
30	28	24	32	30	30	30	34	30	34	30	38	36	36	36	32
31	28	40	24	34	26	26	30	30	34	30	30	24	32	32	28
32	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
33	32	32	32	32	32	32	32	32	32	32	32	32	32	32	32
34	28	30	34	30	34	28	40	28	32	26	38	34	30	16	20
35	32	30	30	38	30	28	36	32	32	30	30	30	38	36	28
36	30	38	36	32	38	30	36	36	26	30	36	32	46	34	32
37	38	34	32	32	38	34	32	28	26	34	24	32	30	38	28
38	34	36	30	30	32	34	28	36	30	28	30	38	32	30	32
39	22	32	30	38	36	38	28	32	38	20	34	34	32	38	28
40	36	30	30	32	36	26	34	34	26	36	32	38	30	28	32
41	32	34	38	32	32	38	34	34	30	24	32	30	26	32	32
42	32	28	24	38	38	38	26	38	34	30	30	24	36	28	36
43	40	32	36	38	30	26	38	34	38	30	38	28	32	36	36
44	34	36	26	32	26	32	38	30	28	34	28	30	36	38	32
45	30	28	30	32	30	24	34	30	32	26	24	30	32	30	36
46	38	34	28	38	36	36	30	22	24	32	30	36	30	34	32
47	38	26	28	38	28	36	30	34	36	36	26	32	34	30	28
48	34	30	32	28	26	30	28	36	34	34	32	32	34	34	36
49	34	30	32	32	30	34	32	32	30	30	28	32	34	34	36
50	38	32	30	30	40	34	36	32	42	32	34	30	36	34	32
51	26	32	42	34	32	30	28	32	38	32	22	34	36	30	32
52	32	20	36	28	32	36	24	28	32	28	32	28	28	32	32
53	24	32	32	40	28	36	32	32	28	28	32	36	36	28	36
54	36	30	26	30	30	40	32	36	28	30	30	38	34	28	32
55	24	26	26	26	38	32	36	44	32	34	30	34	34	36	28
56	34	36	26	32	30	36	30	26	36	26	32	38	36	30	32
57	30	40	34	28	38	28	26	30	28	34	36	30	32	34	32
58	38	22	32	34	36	32	30	38	28	32	34	36	30	30	28
59	30	26	28	22	32	24	30	22	36	36	30	32	34	30	36
60	24	26	30	32	28	34	34	26	34	36	32	42	30	36	36

61	36	34	34	36	36	30	34	30	42	32	32	34	34	36	32
62	28	36	28	34	34	30	34	34	30	30	30	36	28	32	36
63	28	28	28	46	38	26	30	34	30	38	30	32	32	28	32

Таблица 1 Значения $S_5(i, j)$.

Уравнения для эффективных линейных статистических аналогов всех S-боксов приведены в следующей таблице 2. Здесь $\vec{X} = (x_1, \ldots, x_6)$, $\vec{Y} = (y_1, \ldots, y_4)$, $\vec{K} = (k_1, \ldots, k_6)$ - входные, выходные и ключевые вектора соответственно.

№ *S*-бок-са	Эффективное линейное уравнение	p	$\Delta = \lvert 1 - 2p \rvert$
1	$x_2 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_2$	7/32	9/16
2	$x_1 \oplus x_5 \oplus y_1 \oplus y_3 \oplus y_4 = k_1 \oplus k_5$	1/4	1/2
3	$x_1 \oplus x_5 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_1 \oplus k_5$	1/4	1/2
4	$x_1 \oplus x_5 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_1 \oplus k_5$ $x_1 \oplus x_3 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_1 \oplus k_3$ $x_1 \oplus x_3 \oplus x_5 \oplus x_6 \oplus y_2 \oplus y_3 = k_1 \oplus k_3 \oplus k_5 \oplus k_6$ $x_1 \oplus x_3 \oplus x_5 \oplus x_6 \oplus y_1 \oplus y_4 = k_1 \oplus k_3 \oplus k_5 \oplus k_6$	1/4 1/4 1/4 1/4	1/2 1/2 1/2 1/2
5	$x_2 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_2$	3/16	5/8
6	$x_2 \oplus y_2 \oplus y_3 \oplus y_4 = k_2$ $x_1 \oplus x_5 \oplus y_1 \oplus y_3 \oplus y_4 = k_1 \oplus k_5$	9/32	7/16
7	$x_1 \oplus x_2 \oplus x_3 \oplus x_5 \oplus x_6 \oplus y_2 = k_1 \oplus k_2 \oplus k_3 \oplus$ $\oplus k_5 \oplus k_6$	7/32	9/16
8	$x_2 \oplus y_1 \oplus y_2 \oplus y_3 \oplus y_4 = k_2$ $x_1 \oplus x_5 \oplus y_1 \oplus y_2 \oplus y_3 = k_1 \oplus k_5$	1/4	1/2

Таблица 2.

Исходя из результатов таблицы 2, можно сделать вывод, что эффективным линейным статистическим аналогом одного раунда DES является уравнение (10). С учётом функции расширения и перестановки алгоритма DES (см. таблицу 2 п.1.1) имеем эффективный линейный статистический аналог *i*-го раунда DES

$$\vec{Y}(i)[2,8,14,24] \oplus \vec{X}(i)[17] = \vec{K}(i)[26], \qquad (11)$$

с $\Delta = 5/8$.

Мацуи не описывал алгоритма получения эффективных линейных аналогов для *r*-раундового DES, но из финальных результатов можно предположить, что основная идея состоит в следующем: использовать связанные тройные суммы с наибольшим Δ.

Поясним это на примере. Для этого рассмотрим схему 3-раундового DES.

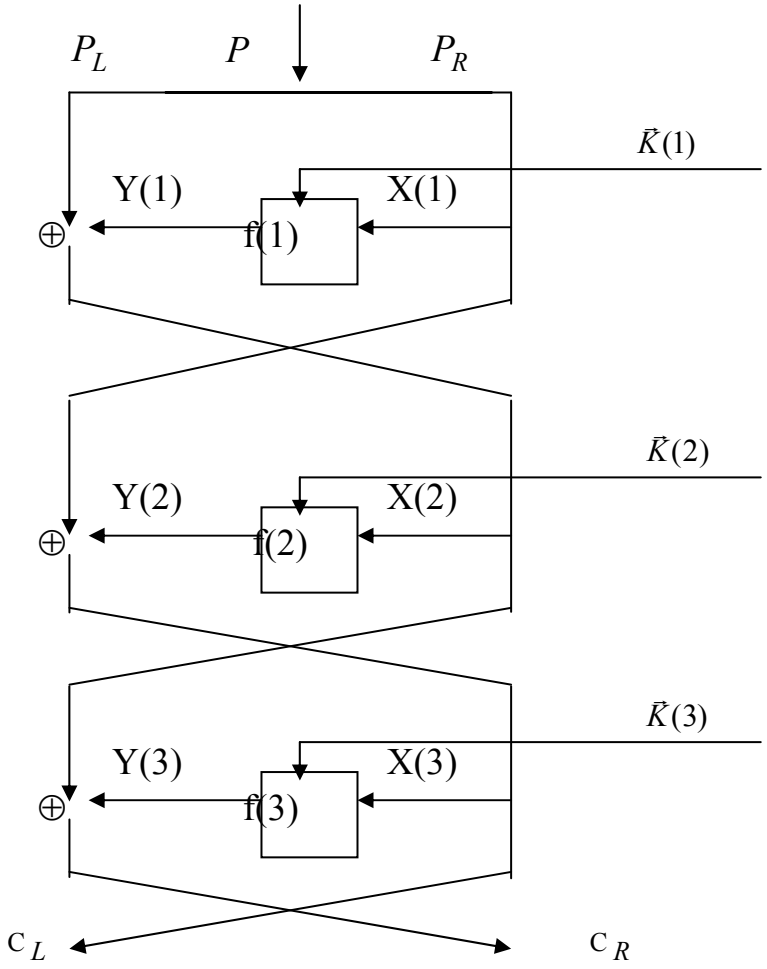


Рис. 1. Схема 3-х раундов DES.

Эффективными линейными статистическими аналогами 1-ой и 3-ей итераций являются уравнения

$\vec{Y}(1)[2,8,14,24] \oplus \vec{X}(1)[17] = \vec{K}(1)[26]$,

$\vec{Y}(3)[2,8,14,24] \oplus \vec{X}(3)[17] = \vec{K}(3)[26]$,

каждое из которых выполняется с вероятностью $3/16$.

Учитывая, что $\vec{X}(1) = P_R$, $\vec{X}(3) = C_L$, $\vec{Y}(1) = P_L \oplus \vec{X}(2)$, $\vec{Y}(3) = C_R \oplus \vec{X}(2)$, после сложения этих уравнений получим

$$(P_R \oplus C_L)[17] \oplus (P_L \oplus C_R)[2,8,14,24] = (\vec{K}(1) \oplus \vec{K}(3))[26]. \qquad (12)$$

По лемме 1 $\Delta = \frac{5}{8} \cdot \frac{5}{8} = \frac{25}{64}$ и это уравнение выполняется с вероятностью

$$\text{p} = \frac{1-\Delta}{2} = \frac{39}{128}.$$

Найти биты ключа $\vec{K}(1) \oplus \vec{K}(3)$ можно, решая уравнение (12) с использованием следующего алгоритма.

Алгоритм. Пусть *N*-число всех открытых текстов и *T*-число открытых текстов, для которых левая часть (11) равна 0.

Если $T > N/2$, то

$$\vec{K}(1) \oplus \vec{K}(3) = \begin{cases} 0, & p > 1/2 \\ 1, & p < 1/2 \end{cases}.$$

Если $T < N/2$, то

$$\vec{K}(1) \oplus \vec{K}(3) = \begin{cases} 1, & p > 1/2 \\ 0, & p < 1/2 \end{cases}.$$

Успех алгоритма возрастает с ростом *N* и $\Delta = |1-2p|$. Вероятность успеха определяется следующей леммой.

Лемма 2. Пусть *N* –число открытых текстов и *p* - вероятность выполнения линейного уравнения (2). Предположим, что $\Delta = |1-2p| \to 0$. Тогда вероятность успеха алгоритма имеет вид

$$P \to \int_{-\sqrt{N}\Delta}^{\infty} \varphi(x)dx,$$

где

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}. \qquad (13)$$

Числовые вычисления (13) сведены в таблице 3.

N	$1/16\Delta^2$	$1/8\Delta^2$	$1/4\Delta^2$	$1/2\Delta^2$
$P_{усп}$	0,841	0,921	0,977	0,998

Таблица 3.

2.10. Дифференциальный криптоанализ.

Пусть некоторый блочный шифратор с длиной блока N строится по схеме.

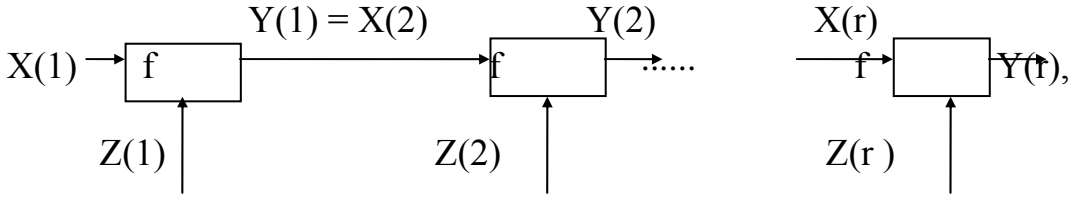


Рис. 1

где $\overline{Z} = (Z(1), Z(2),\ldots,Z(r))$ получается по некоторой схеме из Z_0, или независимо и равновероятно выбирается для каждого цикла. Пусть $X(1)$ и $X^*(1)$ – пара открытых текстов. Рассмотрим следующие разности:

$\Delta X(1) = X(1) - X^*(1)$,

$\Delta Y(i) = Y(i) - Y^*(i)$.

Если мы знаем открытый и шифрованный текст в одном i-ом цикле DES, то несложно определить часть ключа $Z(i)$, которая используется в i-ом цикле (идея Шеннона о суперпозиции простых шифров для получения сложного шифра). Но, если мы знаем $\Delta X(i)$, $Y(i)$ и $Y^*(i)$, то задача становится более сложной, но также решается, если решается предыдущая задача.

Определение 1. Преобразование f называется криптографически слабым, если по $\Delta Y(r\text{-}1)$, $Y(r)$ и $Y^*(r)$ для некоторого (малого) числа пар $(X(1), X^*(1))$ можно найти (хотя бы часть) $Z(r)$.

Идея дифференциального криптоанализа [Biham 93] заключается в том, чтобы найти такие $\Delta X(1)$, что при случайном равновероятном выборе $X(1)$, $Z(1)$, $Z(2),\ldots Z(r\text{-}1)$ с вероятностью более $\frac{1}{2^N}$ появится $\Delta Y(r\text{-}1)$.

Определение 2. Пара (α, β) возможных значений вектора $(\Delta X(1), \Delta Y(i))$ называется дифференциалом i-ого цикла.

Тогда дифференциальный криптоанализ описывается следующей моделью.

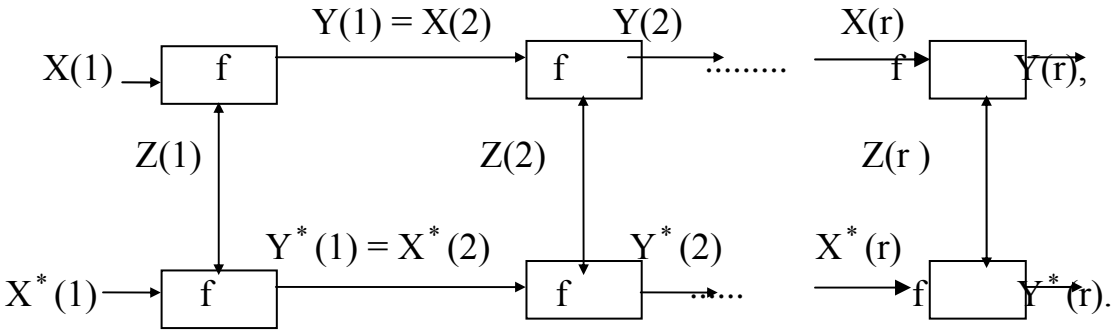


Рис. 2

Пусть f определяет операции в Δ и f криптографически слаба. Тогда возможна следующая атака.

1. Выбираем дифференциал (r-1) – го цикла (α, β), для которого вероятность $P(\Delta Y(r\text{-}1) = \beta \mid \Delta X(1) = \alpha)$ большая.
2. Случайно выбираем $X(1)$ и подбираем $X^*(1)$, чтобы $\Delta X(1) = \alpha$. Пусть известны $Y(r)$ и $Y^*(r)$.
3. Делаем предположение, что $\Delta Y(r\text{-}1) = \beta$ и, зная $Y(r)$ и $Y^*(r)$, находим Z(r).
4. Повторяем 2 и 3 пока один (частичный) ключ не начнет появляться чаще других. Это и будет Z(r).
5. Повторяем 1-4 до нахождения полного ключа.

2.11. Метод коллизий для хэш-функций.

Для контроля целостности сообщений, передаваемых по каналу связи, используется следующий метод. Пусть $A = \{a_1,...,a_m\}$ - алфавит, A^* - множество слов конечной длины в алфавите A. Пусть определена функция $H: A^* \to A^l$, которая обладает тем свойством, что значения функции H на словах, которые даже при отличии друг от друга только в одном знаке дают значительно отличающиеся хэш значения. Тогда, получив на приемном конце сообщение и его хэш, можно вычислить значение хэш от сообщения, сравнить с полученным хэш по каналу связи, и подтвердить или опровергнуть, что сообщение не искажено.

Если функция H зависит также от ключа $k \in K$, то помимо проверки целостности добавление значения хэш к сообщению подтверждает истинность сообщения. Такой способ подтверждения истинности называется кодом аутентификации (Message Autentification Code - MAC). Однако, такое подтверждение истинности еще не является электронной подписью. Подтверждение истины называется подписью, если ее могут проверить все, не знающие ключи. Например, в суде можно поверить истинность, не раскрывая ключи. Приведенный выше способ проверки

подлинности сообщения непригоден, так как влечет раскрытие ключа. Для того, чтобы код аутентификации стал электронной подписью сообщения, необходимо использовать хэш-функции с дополнительными свойствами. Например, использовать систему с открытым ключом. Напомним, что у корреспондента В имеются два алгоритма E_B и D_B, каждый из которых преобразует слово из A^l в слово из A^l и каждый определен на A^l. Первый алгоритм известен всем, а второй - только владельцу В. Обладание алгоритмом D_B однозначно (юридически) определяет корреспондента В. Считаем, что E_B и D_B удовлетворяют соотношениям $\forall\ \alpha \in A^l$

$E_B\ D_B(\alpha) = \alpha,\ D_B\ E_B(\alpha) = \alpha.$

Тогда электронной подписью документа $M \in A^*$ называется
$C = D_B(H(M))$.

Проверка подписи под документом M возможна любым лицом, у которого есть E_B. Для этого проверяющий вычисляет $H(M)$ ($H(\ )$ - известна всем, M проверяющий получает вместе с подписью C). Второй шаг проверки - вычисление

$E_B(C) = E_B\ D_B(\ H(M)) = H(M)$.

Если вычисленное значение хэш от M совпало с результатом применения алгоритмов E_B к C, то подпись В считается подтвержденной (даже в суде).

Свойства хэш-функций.

1. Если $M \neq M'$ хотя бы в одном знаке, то $H(\ M) \neq H(M')$ примерно на половине знаков.

Это свойство отмечалось выше. Отметим также, что могут найтись такие $M \neq M'$, что $H(\ M) = H(M')$, но выбор таких M и M' случайно маловероятен, а подбор труден.

2. Для произвольной строки $\alpha \in A^l$ трудно подобрать $M \in A^*$ такое, что $H(\ M) = \alpha$.

3. $H(\ M)$ вычисляется быстро.

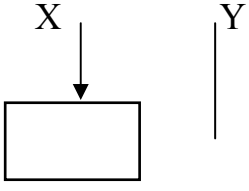
Выполнить п. 1)-3) не просто. В РФ принят стандарт ГОСТ 3411.94 на хэш. Длина хэш 256 бит.

Построение хэш.

Обычно хэш $H(\ )$ строится следующим образом.

1) Выбирается функция h: $A^l \times A^l \to A^l$, удовлетворяющая свойствам 1-3.

Например, когда длина хэш была 64 бита, то брали следующую схему



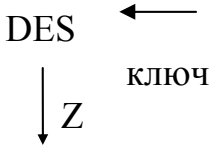


Рис. 1

$z = h(x,y) = \mathrm{DES}_y(x)$.

2) $\forall\ \alpha \in A^*,\ \alpha = \alpha_1 \ldots \alpha_t$,

где длина блока α_i равна l, а блок α_t дополнен до блока длины l(если это необходимо). $H(\alpha)$ вычисляется по следующему алгоритму.

$\beta_1 = h(\alpha_1, \alpha_2)$

$\beta_2 = h(\beta_1, \alpha_3)$

$\beta_{t-1} = h(\beta_{t-2}, \alpha_t)$.

Основная атака на хэш - это метод коллизий. Пусть B - мошенник, а D - лицо, подвергающееся атаке. B готовит два документа M и M' так, что документ M пользователь D охотно подпишет, а B заинтересован в подписи D под документом M'. Тогда, подписав у D документ M, B получает подпись $C = D_B(H(M))$. Если M' выбрано так, что $H(M) = H(M')$, то B предъявляет суду (M', C). Судья убеждается, что подпись под M' принадлежит D. тогда атака удалась.

Каким образом можно выбрать два таких сообщения? По свойству 2 подбор осуществить сложно.

Оказывается, с большой вероятностью злоумышленник может реализовать подбор такого сообщения следующим образом. Такой подбор основан на задаче о днях рождения.

B выбирает два нужных сообщения M и M', при этом D может подписать сообщение M, но не подпишет сообщение M'. Варьируя в сообщениях интервалами, шрифтами, форматами и т.п., B получает n пар вариантов M и M' без изменения их смысла. И, следовательно, n пар соответствующих им хэш-функций.

$$\begin{array}{ll} H(M_1) & H(M_1') \\ H(M_2) & H(M_2') \\ \vdots & \vdots \\ H(M_n) & H(M_n') \end{array} \qquad (1)$$

Затем, просматривая все пары, ищет совпадения. Сообщения $M_1,\ldots,M_n$ отличаются слабо, а их хэш-функции отличаются значительно, так что мы можем считать, что значения хэш-функций выбираются

случайно, равновероятно и независимо друг от друга. Каково должно быть значение n, чтобы такая пара появилась с большой вероятностью? Обозначим $|A^l| = N$ и пусть p – вероятность того, что имеется пара сообщений M_i и $M_i^{'}$ таких, что $H(M_i) = H(M_i^{'})$, то есть вероятность успешной атаки. Нам проще вычислить вероятность противоположного события, то есть, что в (1) нет такой пары.

$$1 - p = \frac{\binom{N}{n}\binom{N-n}{n}}{\binom{N}{n}\binom{N}{n}} = \frac{(N-n)!n!(N-n)1}{n!(N-2n)!N!} = \frac{[(N-n)!]^2}{N!(N-2n)!}. \qquad (2)$$

Условия, налагаемые на n, при которых вероятность (2) мала, приводятся в следующей теореме.

Теорема. *Пусть N, n* $\to \infty$*, но* $\frac{n^2}{N} \to t > 0$*, тогда*

$p = (1 - e^{-t})\,(1 + o(1))$.

Доказательство. Используя формулу Стирлинга, получим:

$$1 - p = \frac{[(N-n)!]^2}{N!(N-2n)!} = \frac{[(N-n)^{N-n}e^{-N+n}\sqrt{2\pi(N-n)}]^2(1+o(1))}{N^N e^{-N}\sqrt{2\pi N}(N-2n)^{N-2n}e^{-N+2n}\sqrt{2\pi(N-2n)}} =$$

$$= \frac{(1-\frac{n}{N})^{2N-2n}}{(1-\frac{2n}{N})^{N-2n}}(1+o(1)).$$

Отсюда, используя разложение логарифма в ряд, получим

$$ln(1-p) = [(2N-2n)ln(1-\frac{n}{N}) - (N-n)ln(1-\frac{2n}{N})](1+o(1)) =$$

$$= [(2N-2n)(-\frac{n}{N} - \frac{n^2}{2N^2} + O(\frac{n^3}{N^3})) - (N-n)(-\frac{2n}{N} - \frac{2n^2}{N^2} +$$

$$+ O(\frac{n^3}{N^3})](1+o(1)) = -\frac{n^2}{N}(1+o(1)) = -t\,(1+o(1)).$$

Таким образом,

$1 - p = e^{-t}(1 + o(1))$,

и

$p = (1 - e^{-t})\,(1 + o(1))$,

что и требовалось доказать.

Пример. Если $N = 2^{64}$, то при выборе $n = \sqrt{N} = 2^{32}$ сообщений, вероятность успешной атаки будет равна $p = (1 - e^{-1})(1 + o(1)) \approx \frac{2}{3}$.

2.12. Анализ схем шифрования, использующих многократно один блочный шифр.

Пусть для определенности используется DES. Достижения в криптографии и электронике в 90-х годах позволили снизить время дешифрования DES, что позволяет делать недорогие системы для его дешифрования. Возникла идея построения нового алгоритма, у которого стойкость выше, чем у DES, а в качестве компоненты алгоритма используется DES. Наибольшую потребность в этом испытывают финансовые структуры, которые не хотели бы отказываться от DES из-за значительных дополнительных вложений и других трудностей, но нуждаются в стойких системах шифрования. В этих структурах наиболее употребительным является режим с зацеплением блоков (CBC).

Режимы использования DES [Men., с.229]**.**

1. Режим электронной книги (ECB).

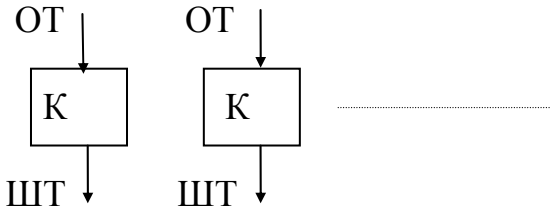


Рис.1

2. Режим с зацеплением (CBC).

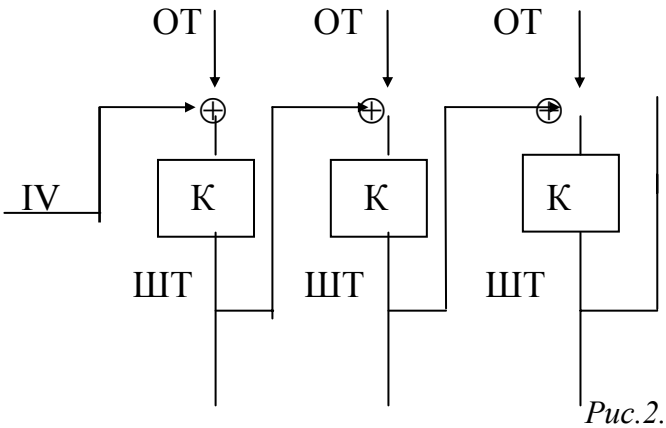


Рис.2.

Здесь IV – инициальное значение.

3. Режим с обратной связью по шифртексту (CFB).

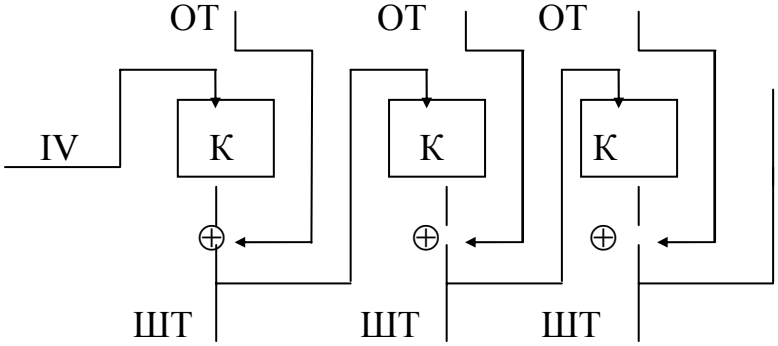


Рис. 3.

4. Режим с обратной связью (асинхронный) ((OFB).

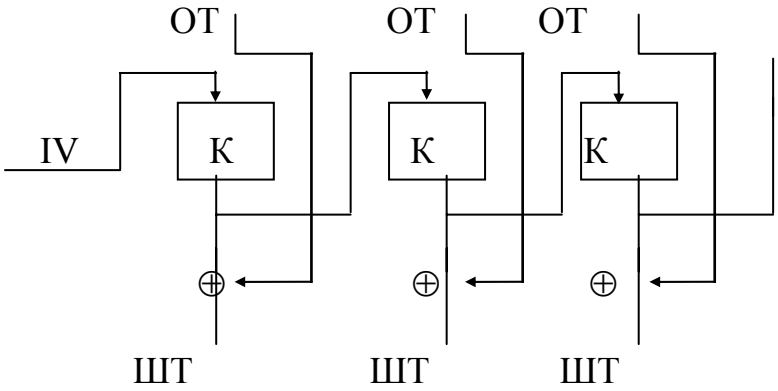


Рис. 4.

Были придуманы схемы шифрования, многократно использующие DES.

Пример 1. Двойной DES в режиме электронной книги.

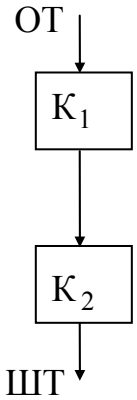


Рис. 5.

Двойной DES не стоек к методу “встреча посередине”.

Пример 2. Тройной DES в режиме электронной книги [Men., с.272]. Стойкость данного алгоритма не превосходит стойкости DES, если используется атака с выделенным открытым текстом.

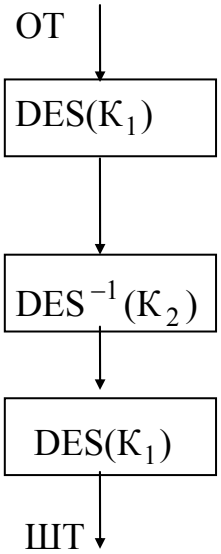


Рис. 6.

Пример 3. Тройной DES с зацеплением.

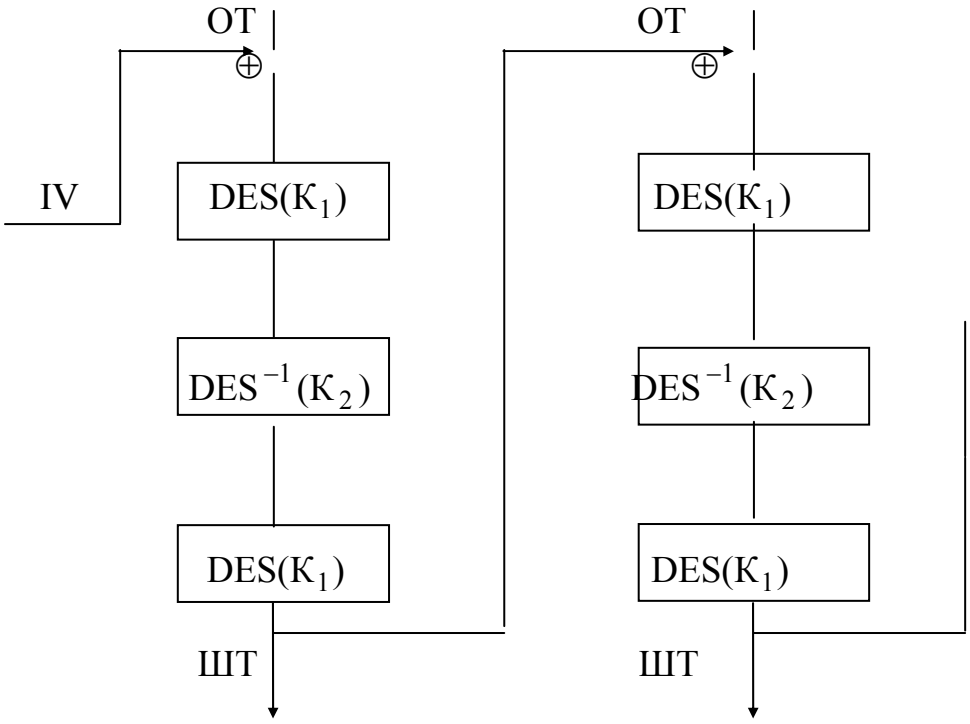


Рис. 7.

Оказывается, что стойкость данного тройного DES такая же как стойкость в режиме электронной книги при атаке с выбранным шифртекстом. Слабость данного алгоритма заключается в том, что

шифртекст известен и он же идет на вход следующего блока.

Пример 4 [Biham 94]. Возникло много модификаций тройного DES, когда зацепление проходит внутри. Рассмотрим следующий вариант тройного DES: CBC/CBC/ECB.

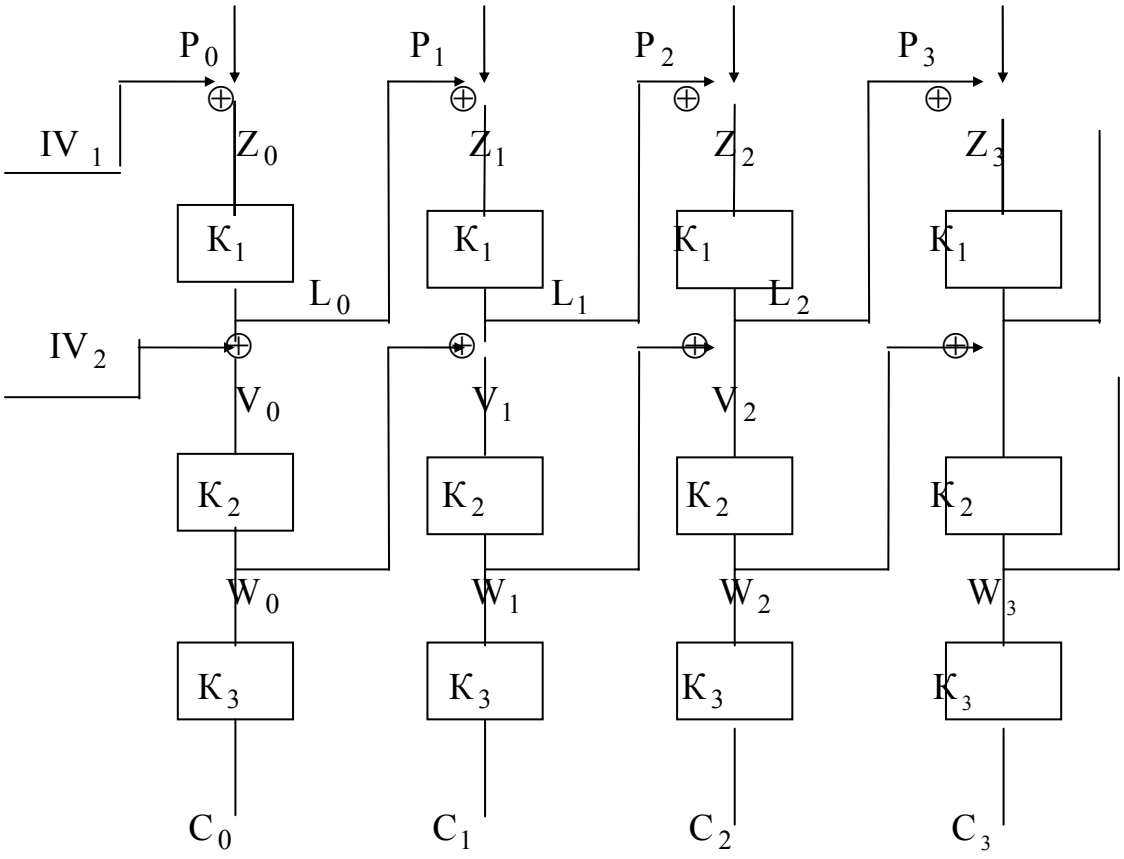


Рис. 8.

Рассмотрим один из вариантов дешифрования этой схемы методом выбранного шифртекста. Сначала будем искать ключ K_3. Пусть известны открытые и шифрованные тексты. Среди всех шифртекстов выберем две тройки блоков следующего вида (C_0, C_1, C_2) и (C_0^*, C_1, C_2), где $C_0 \neq C_0^*$. Обозначим для первой тройки через W_i - вход последнего блока, V_i - вход второго блока и через Z_i, L_i - вход и выход первого блока соответственно, P_i - открытый текст. Соответствующие обозначения для второй тройки – $W_i^*, V_i^*, Z_i^*, L_i^*, P_i^*$. Для выбранных троек имеем следующие соотношения. Из того, что $C_1 = C_1^*$, $C_2 = C_2^*$, следует

$$W_1 = W_1^*,\ W_2 = W_2^*,\ V_1 = V_1^*,\ V_2 = V_2^*. \qquad (1)$$

Отсюда $L_2 = L_2^*$, и, следовательно, $Z_2 = Z_2^*$. Тогда

$$V_1 = W_0 + L_1,$$

$$V_1^* = W_0^* + L_1^*.$$

Из этих соотношений с учетом (1) получим

$$W_0 + W_0^* = L_1 + L_1^*. \qquad (2)$$

Аналогично,

$$Z_2 = P_2 + L_1,$$

$$Z_2^* = P_2^* + L_1^*.$$

Откуда следует, что

$$P_2 + P_2^* = L_1 + L_1^*. \qquad (3)$$

Из соотношений (2) и (3) получим

$$P_2 + P_2^* = W_0 + W_0^*. \qquad (4)$$

Так как мы знаем открытый текст, то нам известно $P_2 + P_2^*$. Тогда мы знаем $W_0 + W_0^*$ и C_0, C_0^* и можем опробовать ключ K_3, что в среднем потребует 2^{55} операций.

Нахождение вероятностей таких пар троек, у которых $C_1 = C_1^*$ и $C_2 = C_2^*$ следует из задачи о днях рождения. А именно, если вероятность такой пары при произвольном C_0 есть $\frac{1}{2^{128}}$, то вероятность хоть какой-нибудь пары $\sim \sqrt{\frac{1}{2^{128}}} = \frac{1}{2^{64}}$. Значит надо иметь $\sim 2^{64}$ блоков шифртекста для определения ключа K_3.

Зная K_3, находим W_i и, применяя аналогичную технику, определяем K_2 и K_1. Для данной атаки потребуется $3 \cdot 2^{55}$ операций опробования и $3 \cdot 2^{64}$ блоков шифртекста.

2.13. Атака на тройной DES с помощью линейного криптоанализа.

Рассмотрим схему тройного DES в режиме CBC/ECB/CBC [Biham 94].

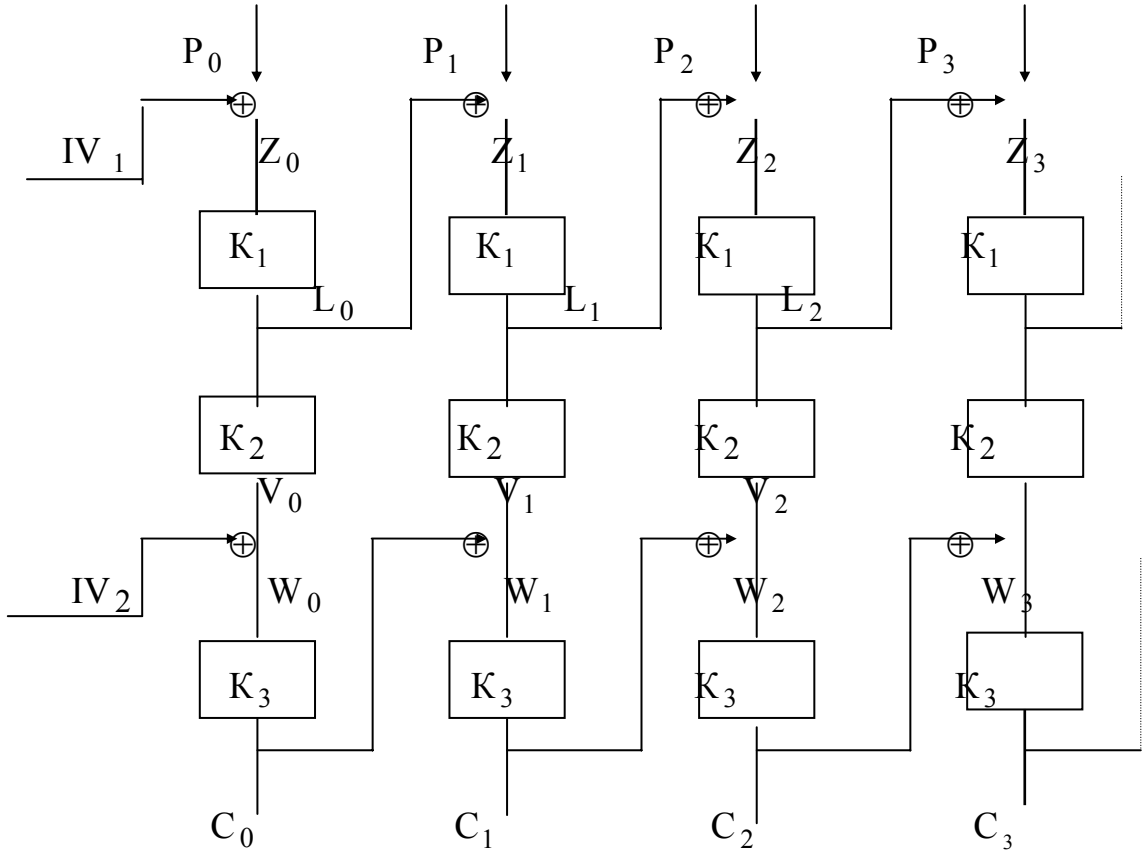


Рис. 1.

Пусть найдено достаточно много шифртекстов (C_0, C_1, C_2), где C_1 и C_2 - фиксированы, а C_0 - произвольные и различные.

1) Сначала атакуем ключ K_2. Обозначим через W_i - вход последнего блока, V_i - выход второго блока и через Z_i, L_i - вход и выход первого блока соответственно, P_i - открытый текст. Для всех векторов

$$L_1 = D_{K_2}(V_1),\ V_1 = C_0 + W_1.$$

Отсюда

$$L_1 = D_{K_2}(C_0 + W_1). \qquad (1)$$

Для всех векторов $V_2 = W_2 + C_1$ одно и то же, так как $W_2 = D_{K_3}(C_2)$. Следовательно, для них $L_2 = D_{K_2}(V_2)$ одно и то же. Отсюда получаем

$$P_2 = D_{K_2}(C_0 + W_1) + Z_2, \qquad (2)$$

где для всех C_0 вектора W_1 и Z_2 остаются одинаковыми. Нам известны P_2 и C_0, а неизвестны K_2, W_1 и Z_2. Пусть $D^*_{K_2}(\cdot)$ – линейный статистический аналог = $D_{K_2}(\cdot)$. Тогда

$$P_2 = D^*_{K_2}(C_0 + W_1) + Z_2 \quad - \qquad (3)$$

- линейное уравнение относительно K_2, W_1 и Z_2. Набирая и решая системы при различных (C_0, P_2), статистически выделяем решение K_2, W_1 и Z_2.

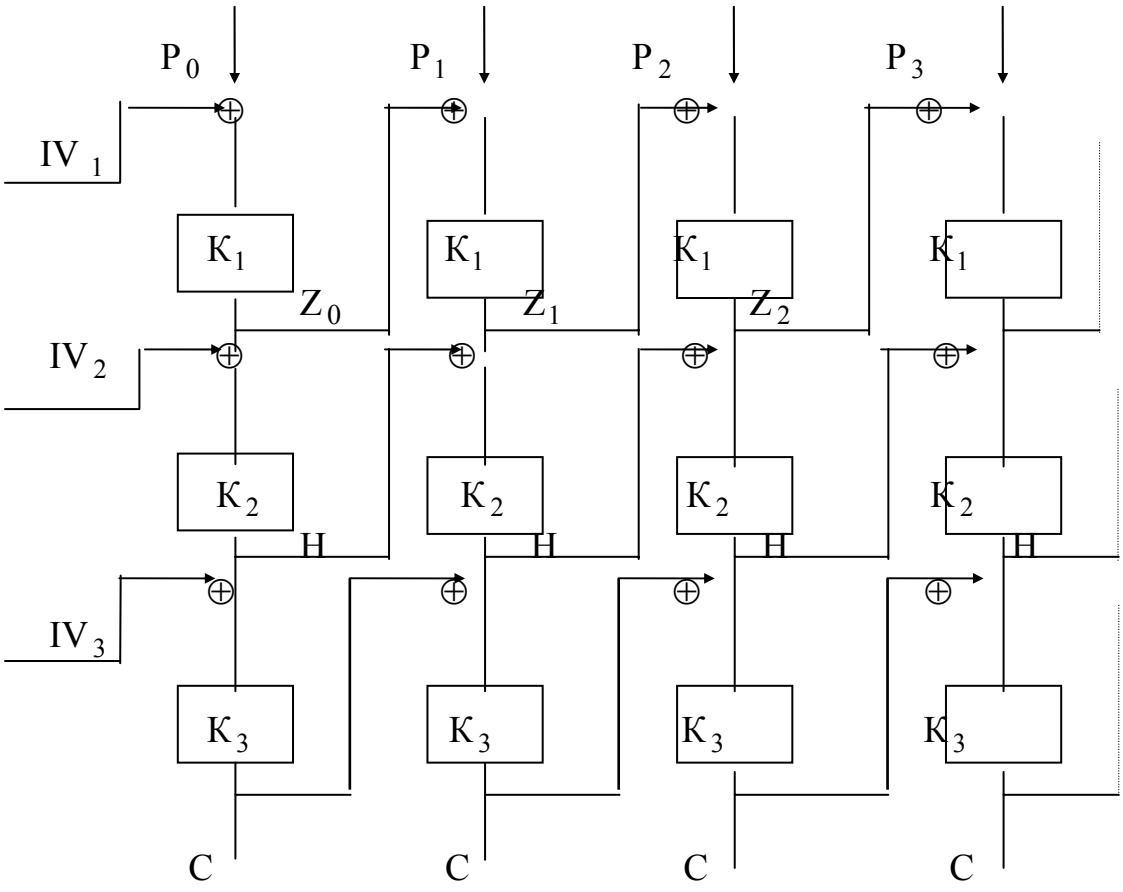
2) $E_{K_3}(W_1) = C_1$.

Отсюда с помощью метода полного перебора получим K_3.

3) Зная K_2, K_3, получим L_2 и, зная Z_2, методом полного перебора из соотношения $E_{K_1}(Z_2) = L_2$ находим K_1.

2.14. Техника атаки на тройной DES, основанная на задаче о днях рождения.

Рассмотрим схему тройного DES CBC/CBC/CBC [Biham 94].



Пусть найдено 2^{33} шифртекстов вида (C, C, C, C). Атака сначала ведется на ключ K_3. На выходе второго блока шифрования имеем четверку

вида (?, *H, H, H*), где

$$H = C + D_{K_3}(C).$$

H - это случайная функция от C, не являющаяся подстановкой. По результатам задачи о днях рождения с большой вероятностью существуют C и C^*, дающие одно и то же H, так как H – не взаимно-однозначное отображение. Тогда для двух C и C^* появится одно и то же P_3. Действительно, для C и C^*

$Z_2 = D_{K_2}(H) + H.$

Отсюда

$$P_3 = Z_2 + D_{K_1}(Z_3) = Z_2 + D_{K_1}(D_{K_2}(H) + \text{H})).$$

Для такой пары C и C^* имеем уравнение

$$C + D_{K_3}(C) = C^* + D_{K_3}(C^*),$$

где C и C^* известны. Методом полного перебора находим ключ K_3. Аналогичным образом находим K_2 и K_1.

2.15. Криптоанализ режима DES, предложенного в качестве стандарта ANSI X9.52.

Для повышения стойкости тройного DES было предложено усилить режим с зацеплением тройного DES маскировками (CBCM). Biham и Knudsen [Biham 98] построили атаку на эту систему, значительно снижающую стойкость. Поэтому данный алгоритм в последний момент был снят с голосования по принятию его как стандарта.

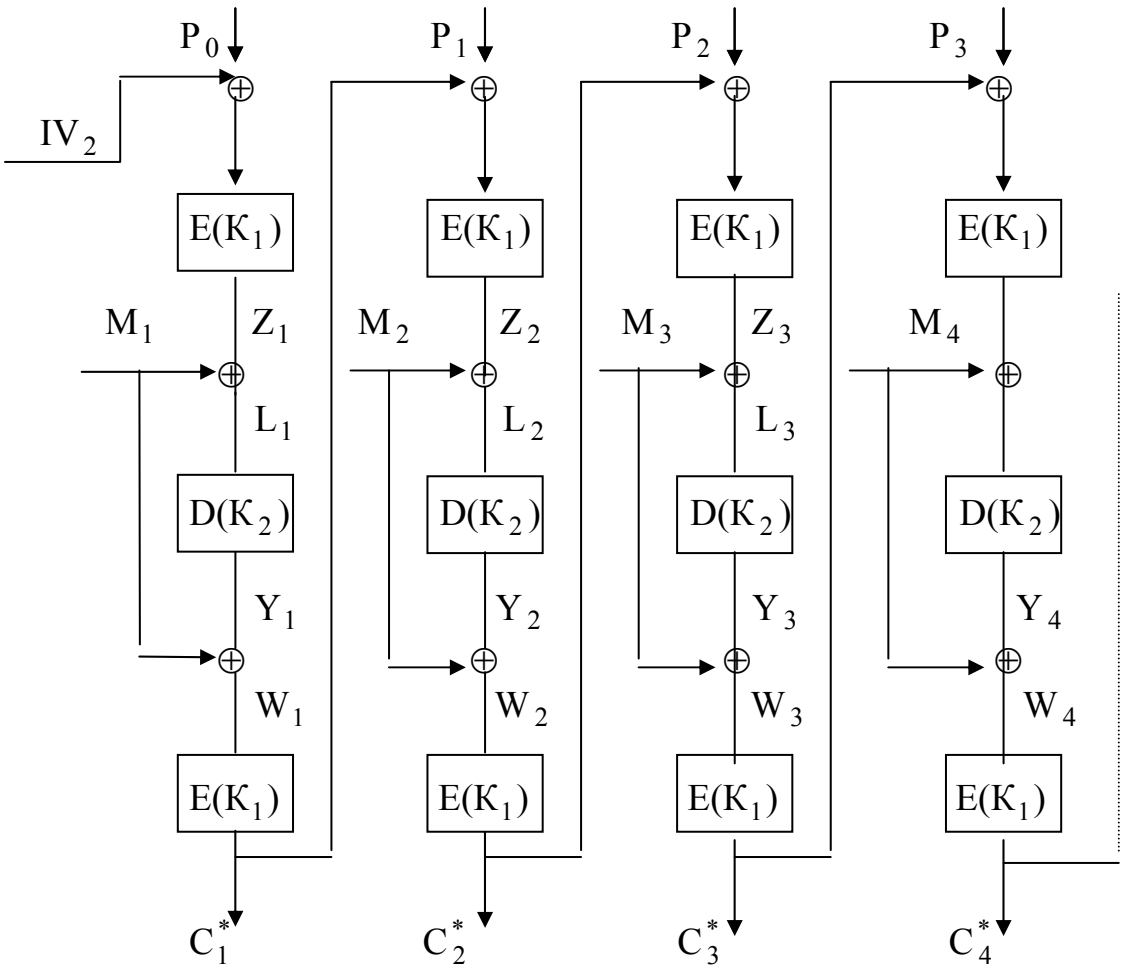


Рис.1.Схема CBCM.

Схема маскировки выглядит следующим образом (OFB).

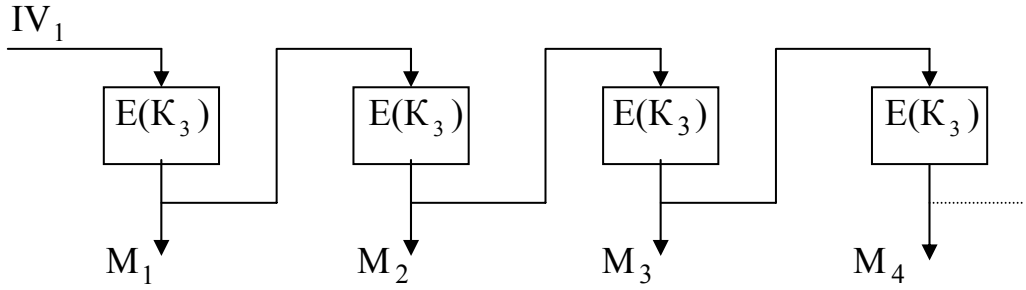


Рис. 2.

Рассмотрим два блока шифртекста длины 2^{64} C_1 и C_2. Ключи K_1, K_2, K_3 и произвольные IV неизвестны. Допустим, что дан такой открытый текст , что получено 2^{64} C_1, а затем 2^{64} C_2. Период схемы OFB самое большее 2^{64} (в среднем 2^{63}). Поскольку IV_2 неизвестно, то не будем принимать во внимание первый блок C_1 и первый блок C_2. Обозначим блоки открытого текста (после удаления первых блоков) в первых p шагах через $P_{1,1}$, ..., $P_{1,p}$. Тогда входы на первые блоки шифрования DES на ключе K_1 равны

$$Q_{1,i} = P_{1,i} + C_1, \ i = 1, ..., p.$$

$Q_{1,i}$ зашифровываются в C_1 при использовании маскирующего блока M_i. Аналогично для любого i $P_{2,i}$ - блок открытого текста, который под действием той же маскировки M_i переводится в C_2.

$$Q_{2,i} = P_{2,i} + C_2, \ i = 1, \ldots, p.$$

Опробуем ключ K_1. Если мы угадали ключ, то должно выполняться равенство

$$E_{K_1}(P_{1,i} + C_1) + D_{K_1}(C_1) = L_i + Y_i \ i = 1, \ldots, p.$$

Аналогичное равенство выполняется и для C_2 :

$$E_{K_1}(P_{1,i} + C_2) + D_{K_1}(C_2) = L_i + Y_i \ i = 1, \ldots, p.$$

Рассмотрим отображение

$$E_{K_2}(\cdot) + V,$$

где V – фиксировано. Это отображение пространства 64-мерных выборок в себя. Пусть V - случайно. Тогда с большой вероятностью существует ровно одна неподвижная точка этого отображения. Эта задача может быть проинтерпретирована следующим образом. Пусть $N = 2^{64}$ дробинок размещают по 2^{64} ящикам. Какова вероятность того, что ровно одна дробинка попадет в свой ящик? N^N общее число размещений N дробинок по N ящикам. Тогда число благоприятных событий (ровно одна дробинка попадет в свой ящик) будет равно N и, соответствующая вероятность

$$p = \frac{N(N-1)^{N-1}}{N^N} = (1 - \frac{1}{N})^N \sim \mathrm{e}^{-1}.$$

Если для некоторого i эта точка равна $M_i + D_{K_1}(C_{K,i})$, то тогда

$$Q_{K,i} = D_{K_1}(M_i + E_{K_2}(M_i + D_{K_1}(C_{K,i}))) =$$
$$= D_{K_1}(M_i + V + (M_i + D_{K_1}(C_{K,i}))) =$$

$= D_{K_1}(V + D_{K_1}(C_{K,i}))$.

Тогда атака состоит из следующих шагов:

1. Выбираем произвольный блок V.
2. Опробуем ключ K_1. Пусть K' – очередной проверяемый ключ.
3. Вычисляем

$$T_1(K') = D_{K'}(V + D_{K'}(C_1)), \quad (1)$$

$$T_2(K') = D_{K'}(V + D_{K'}(C_2)).$$

Находим подходящие тексты такие, что

$Q_{1,i} = T_1(K'), Q_{2,j} = T_2(K')$.

4. Если нашли такие открытые тексты, то вычисляем

$U = D_{K'}(C_1) + D_{K'}(C_2)$.

С учетом (1) последнее равенство влечет

$U = E_{K'}(Q_{1,i}) + E_{K'}(Q_{2,j})$.

Теперь проверяем равенство

$U = E_{K'}(Q_{2,i}) + E_{K'}(Q_{1,j})$.

5. Если равенство выполняется, то ключ $K_1 = K'$. Если равенство не выполняется, то опробуем новый K'.
6. Если ключ K_1 не найден, то опробуем другое V и повторяем действия 1-5.

Объясним причину работы атаки. Поскольку неподвижная точка одна, то

$M_i + D_{K'}(C_1) = M_j + D_{K'}(C_2)$.

Тогда

$$D_{K'}(C_1) + D_{K'}(C_2) = M_i + M_j = U. \quad (2)$$

Рассмотрим теперь $Q_{1,j}$ и $Q_{2,i}$. Для них соответствующие маскировки M_j и M_i, а выходы из блока расшифрования на ключе K_2 равны

$D_{K'}(C_1) + M_j$ и $D_{K'}(\mathrm{C}_2) + \mathrm{M}_i$.

Отсюда, согласно (2),

$Y_{1,j} = D_{K'}(C_1) + M_j = D_{K'}(C_2) + M_i = Y_{2,i}$.

Раз равны выходы, то равны и входы:

$L_{1,j} = L_{2,i}$

Тогда

$$Z_{1,j} = M_j + L_{1,j}$$

$$Z_{2,i} = M_i + L_{2,i}.$$

Сложив два последние равенства, получим

$$E_{K'}(Q_{1,j}) + E_{K'}(Q_{2,i}) = Z_{1,j} + Z_{2,i} = M_i + M_j = U.$$

Если $K_1 \neq K'$, то последнее равенство не выполняется.

Ключ K_2 затем находится проще. Общая сложность атаки 2^{58} при известных выбранных шифртекстах длины $2 \cdot 2^{64} = 2^{65}$.

Глава 3. Синтез криптоалгоритмов.

3.1. Синтез поточных шифров.

Пусть открытый текст - последовательность букв $x = x_1 x_2 ....$, ключ - последовательность символов $z = z_1 z_2 ...$, а шифртекст - последовательность букв $y = y_1 y_2 ....$, где $y_i = E(x_i, z_i)$ - функция зашифрования.

Часто последовательность z получается с помощью генератора, то есть автономного автомата. При этом стойкость оценивается следующими способами [Ru.]:

1. С помощью теоретико-информационного подхода.

2. С помощью практического подхода. Пусть $M_1,...,M_n$ - известные методы криптоанализа и для определения ключа по открытому и шифрованному тексту с помощью метода M_i требуется не более N_i операций. Тогда стойкость поточного шифра N оценивается

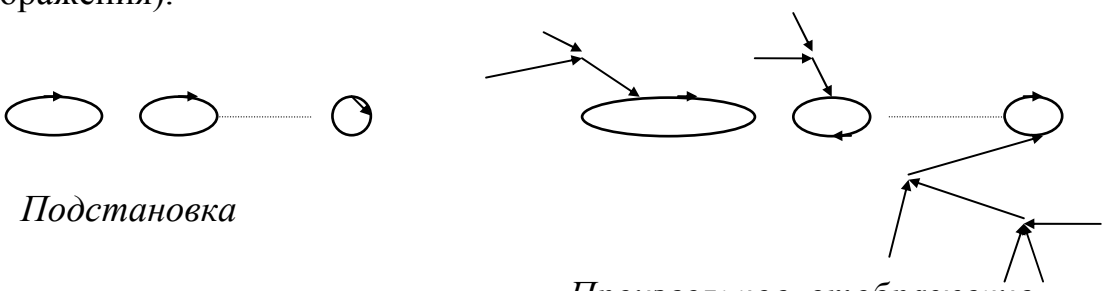
$$N \le \min_{i=1,...,n} N_i .$$

3. С помощью нижней оценки (доказанная стойкость), то есть с помощью оценки снизу для средней трудоемкости по всем методам.

Таким образом, для стойкости шифра против известных методов надо сконструировать такое устройство, которое дает хорошие характеристики по всем методам анализа.

Построим пример синтеза такой системы.

1) Любой генератор z обладает периодом (структура произвольного отображения).



Подстановка

Произвольное отображение

Тогда любой период - попадание на цикл структуры. Если мы попали на малый период, то тогда в исходной гамме появился повтор. Как показано ранее, это слабость, приводящая к дешифрованию. Следовательно, соответствующее устройство должно обладать максимальным периодом. Число циклических точек произвольного отображения не больше числа циклических точек подстановки. Таким образом, в подстановке меньше шансов на короткий цикл. Наилучший вариант, когда цикл в подстановке

единственен, тогда все точки лежат на нем и повторение произойдет не скоро. Оказывается, что регистр сдвига с линейной обратной связью (РСЛС) при определенных многочленах (примитивных) состоит из двух циклов длины 1 и длины 2^n - 1 (полноцикловые РСЛС).

2) Гамма должна быть по статистическим свойствам приблизительно равновероятна, иначе можно применить метод зигзагообразного чтения. Таким образом, если РСЛС полноцикловая, то распределение знаков должно быть приблизительно равновероятным.

3) Однако РСЛС нестойка к аналитическим методам (линейная система). Следовательно, необходимо, чтобы выходная гамма была нелинейной функцией от текущего вектора в регистре. При этом, если f - неравновероятна, то и гамма - неравновероятна. Отсюда функция f должна быть равновероятной.

Построим пример, удовлетворяющий требованиям 1)-3).

Пример 1. $f(x_1,x_2,x_3) = x_1 + x_2x_3$

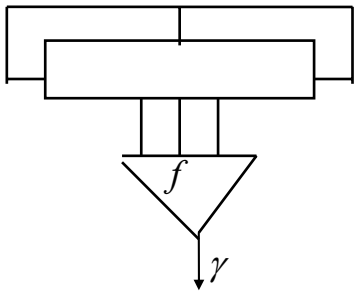


Рис. 1

Действительно,

x_1	x_2	x_3	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$$P(f=0) = P(f=1) = \frac{1}{2}.$$

4) Однако, схема на рис. 1 подвержена корреляционным атакам, следовательно, надо выбирать большое n и функцию f выбирать таким образом, чтобы ее трудно было аппроксимировать линейной рекуррентой для корреляционной атаки (и метода “разделяй и побеждай”).

5) При n большом и функции f с высокой степенью нелинейности получается большой линейный профиль и, следовательно, корреляционная атака не эффективна.

6) Схема на рис. 1 не подвержена атаке “встреча посередине”.

Из условий 1)-6) следует, что схема на рис. 1 - хорошая схема поточного шифра. Однако, против 4) эти схемы все-таки слабы.

Рассмотрим и другие генераторы поточных шифров.

Пример 2. Линейный генератор, управляемый часами [Goll.].

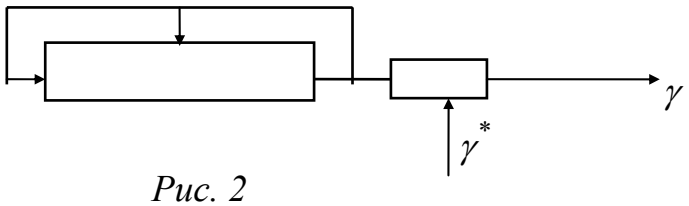


Рис. 2

В этой схеме выбираются только те знаки γ, которые соответствуют единице в последовательности γ^*. Для таких шифров также построены успешные корреляционные атаки.

Пример 3. Генератор Шнорра [Schnorr 88].

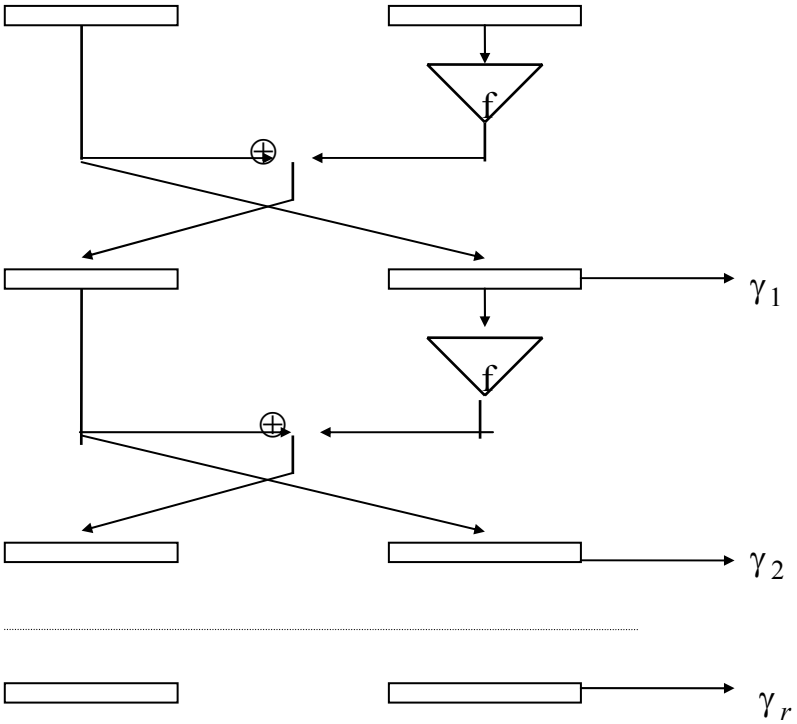


Рис. 3

Сложность восстановления $f \sim n2^n + O(n)$.

3.2. Синхронизация поточных шифров.

Рис. 1

В этой схеме гамма должна вырабатываться синхронно. Если происходит сбой, то рассматриваются две схемы: перезапуск и самовосстановление.

1. Автономный поточный шифр (требует перезапуска при сбое).

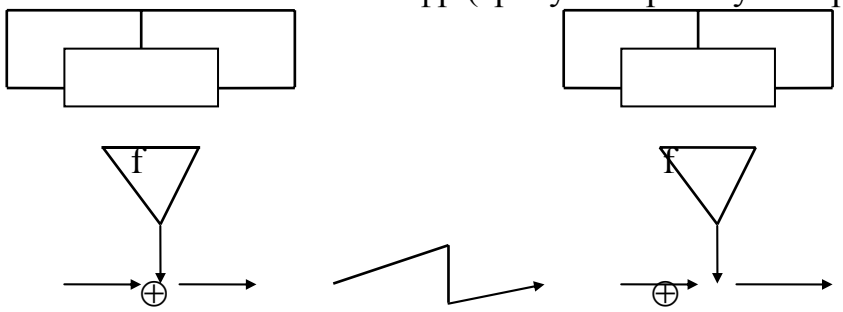


Рис. 2

В этой схеме функция *f* зависит от ключа, а начальное заполнение РСЛС передается по каналу для синхронизации шифров при перезапуске.

2. Самовостановление.

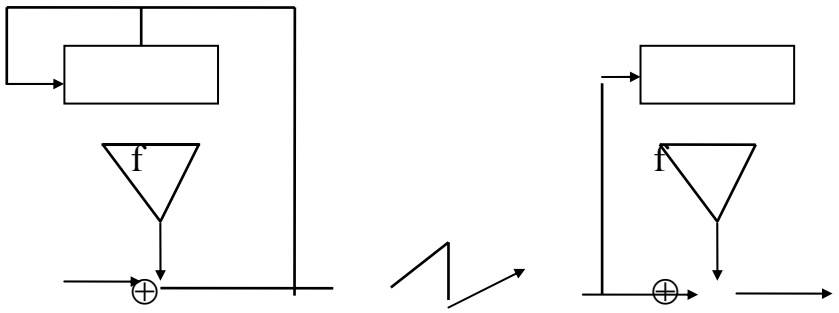


Рис. 3

При передаче и приеме содержимое регистров заполняется ширтекстом. Функция *f* зависит от ключа. При сбое бита в канале расшифрования не происходит, пока искаженный бит находится в регистре. После того, как искаженный бит покидает регистр, правильное расшифрование восстанавливается автоматически. Таким образом, происходит самовосстановление синхронной работы шифраторов на передающем и принимающем узлах.

3.3. Синтез блочных шифров.

Пусть по-прежнему преобразование $Y=T(x,k)$ должно давать трудоемкость восстановления K не ниже порога α по всем известным методам криптоанализа блочных шифров.

Основная идея достигнуть этого (по Шеннону) - итерация простых шифров. Однако, надо иметь в виду, что не всегда суперпозиция простых шифров дает сложный шифр.

Определение 1. Шифр обладает свойством *C*, если статистические свойства шифртекста "сложно" зависят от статистических свойств открытого текста.

Определение 2. Шифр обладает свойством *D,* если

а) каждый знак открытого текста (ОТ) влияет на все знаки шифртекста

б) каждый знак ключа влияет на знаки шифртекста (ШТ).

Необходимыми условиями стойкости блочных шифров являются свойства С и *D*.

Рассмотрим два простых шифра: простую замену φ и перестановку π.

1. Статистические характеристики шифртекста после простой замены равны по значениям статистическим характеристикам открытого текста. Однако, если выписать значения вероятностей любой характеристики шифртекста через статические характеристики открытого текста, то необходимо использовать φ. Таким образом, сложность вычисления статистических характеристик шифртекста не проще сложности вычисления φ. Следовательно, простая замена может порождать свойство *C*.

2. Вероятности знаков шифртекста после перестановки π совпадают с вероятностями знаков открытого текста. Поэтому их вычисление просто. Однако другие статистические характеристики шифртекста (биграммы и т.д.) зависят от многих характеристик открытого текста (если открытый текст не является последовательностью независимых испытаний). Таким образом, перестановка может порождать свойство *Д*.

Следовательно, основная идея построения стойкого блочного шифра - композиция i (i=1,2,...,r) пар простых замен φ_i (с нелинейным преобразованием φ_i) и перестановок π_i.

По виду композиции различают два вида блочных шифров: каскадные шифры и произведение шифров.

а) Каскадный шифр представляет собой композицию простых шифров, когда ключ каждой операции выбирается независимо. Такие шифры подвергаются атаке "встреча по середине" (если ключи малы, то возможна атака "встреча по середине", если ключи большие, то общий ключ очень

большой).

б) Произведение шифров. Выбор ключей для каждой компоненты произведения производится по некоторому алгоритму из одного основного.

Например, DES - произведение шифров.

Определение 3. Инволюцией называется преобразование f такое, что $f=f^{-1}$, т.е. $\forall x \ f(f(x))=x$.

Например,

1) $f(x)= c - x \pmod m$

$$f(f(x))= f(x) + c = - (- x+c)+ c = x.$$

2) Перестановки:

$f(x_1, x_2, x_3, x_4)=(x_3 x_4 x_1 x_2)$.

$f(f(x_1, x_2, x_3, x_4)=f(x_3, x_4, x_1, x_2)=(x_1 x_2 x_3 x_4)$.

Блоки для построения удобного шифрования, расшифрования [Massey 94].

1) Инволюции-перестановки:

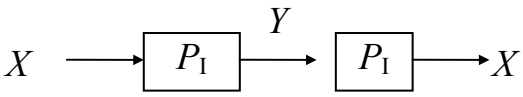
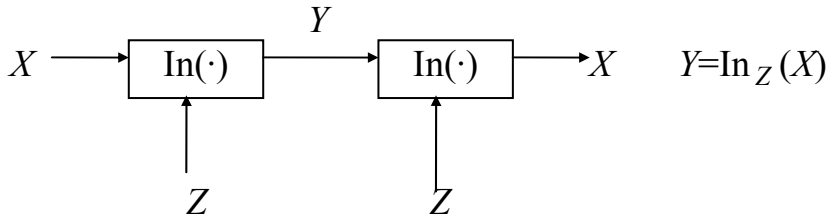


Рис. 1.

2) Инволюции:



$Y=\mathrm{In}_Z(X)$

Рис. 2.

3) Групповые шифры (например, гамма):

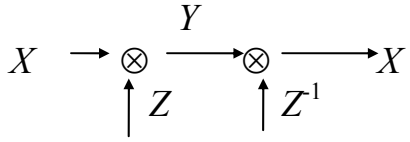
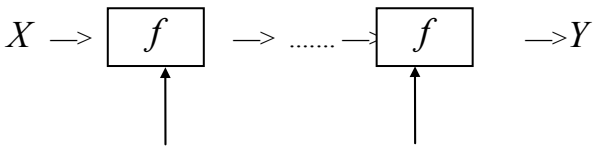


Рис.3.

Здесь $Y=X \otimes Z$ - операция в группе, Z^{-1} - обратный элемент к Z в группе.

Можно получить совершенный шифр для одноразового равновероятного Z .

Утверждение 1. Если использовать инволюции в блоках, то шифрование/расшифрование отличаются порядком ключей.



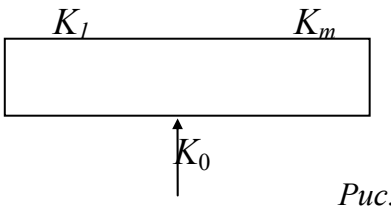


Рис. 4. Шифрование.

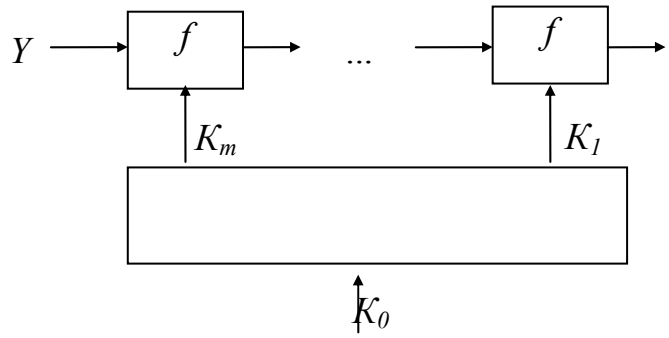


Рис. 5. Расшифрование.

Наиболее часто используются три способа построения блочного шифра:

а) Последовательность: инволюция, инволюция - перестановка (P_I). Примеры: DES; FEAL; LOKI.

б) Чередование группового шифра и инволюции. (при обращении надо брать Z^{-1}_i). Пример: PES.

в) Чередование группового шифра, инволюции, инволюции - перестановки, так что выполняется соотношение

$P_I(a \otimes b)=P_I(a) \otimes P_I(b)$

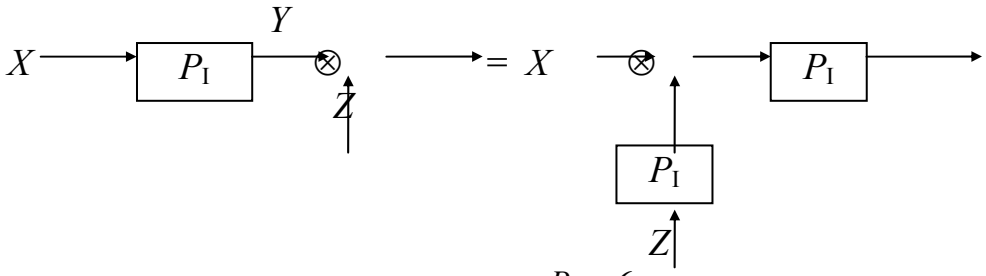
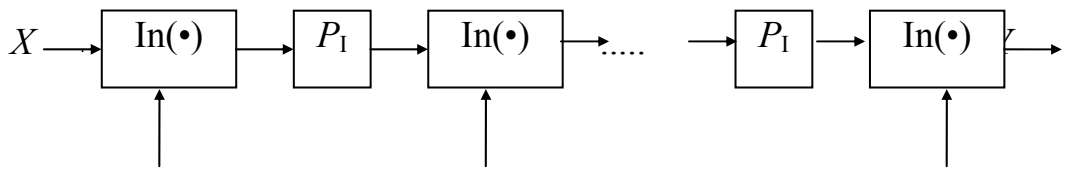


Рис. 6.

Действительно, $P_I(x \oplus P_I(z))=P_I(x) \otimes P_I(P_I(x))=P_I(x) \otimes Z$.

Пример: IDEA



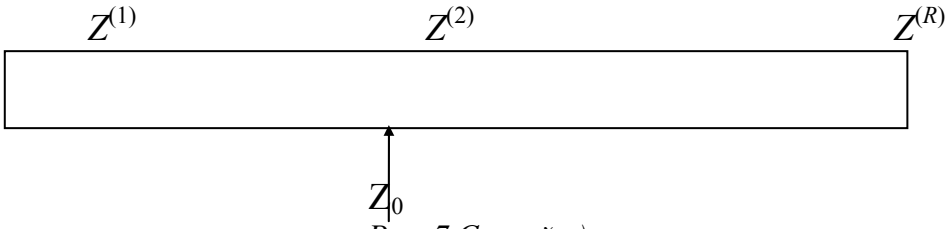


Рис. 7.Случай а).

При расшифровании изменяется порядок ключа.

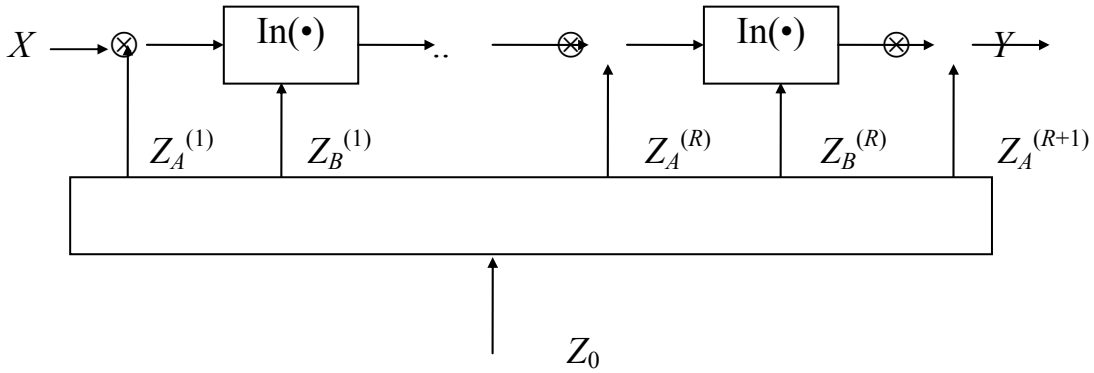


Рис. 8. Случай б).

При расшифровании изменяется порядок ключей и каждый A-подключ заменяется на обратный в группе.

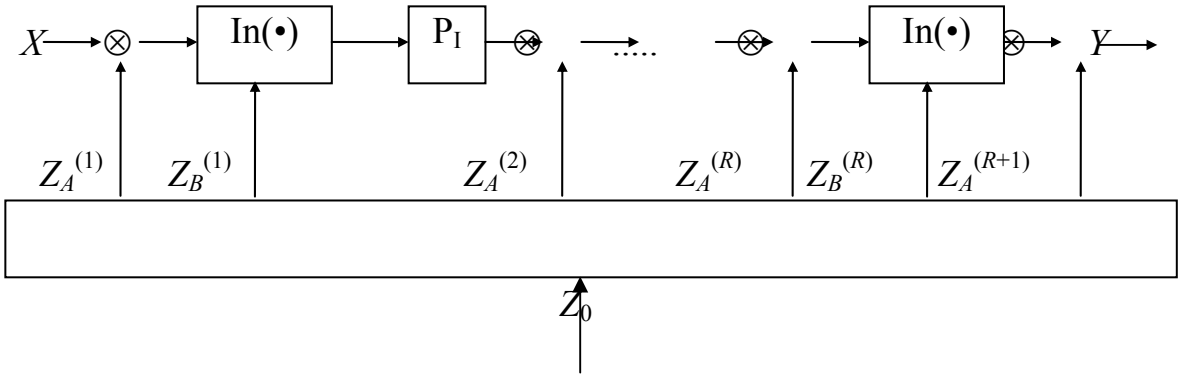


Рис. 9. Случай в).

При расшифровании изменяется порядок ключей, ключи $Z_A^{(1)}$ и $Z_A^{(R+1)}$ заменяются на обратные в группе, а все остальные A-подключи заменяются на обратные с перестановками следующим образом:

$$P_I(Z^{-1})=(P_I(Z))^{-1}.$$

3.4. Слабости блочных щифров.

. Для блочных шифров характерны следующие слабости, которые мы иллюстрируем на примере DES [Massey 94]:

1. Морфизм DES.

Пусть X- входной вектор , а $\overline{X}$ - его отрицание.

Теорема 1. $\mathrm{DES}(\overline{X},\overline{Z})=\overline{\mathrm{DES}(X,Y)}$.

Доказательство. Из отрицания $\overline{Z}$,ключа Z следует, что все 16 производных подключей $\overline{Z}_i$,берутся с отрицанием. Если вместо R_{i-1} и Z_i в алгоритме DES взять $\overline{R}_{i-1}$ и $\overline{Z}_i$,

$f(R_{i-1},Z_i)=f(\overline{R}_{i-1},\overline{Z}_i)$,

тогда при использовании $\overline{R}_{i-1}$, $\overline{L}_{i-1}$ и $\overline{Z}_i$ получим отрицание L_i и R_i

$\overline{L}_i=\overline{R}_{i-1}$ и

$R_i'=\overline{L}_{i-1}\oplus f(\overline{R}_{i-1},\overline{Z}_i)=L_{i-1}\oplus 1\oplus f(R_{i-1},Z_i)=L_{i-1}\oplus f(R_{i-1},Z_i)=\overline{R}_i$.

Теорема доказана.

Замечание 1. Слабость используется при опробовании.

2. Слабые ключи DES.

Определение 1. Ключ K называется слабым ключом DES , если

$\mathrm{DES}_k(\bullet)=\mathrm{DES}_k^{-1}(\bullet)$,

то есть

$Y=D_k(X)$ и $X=D_k(Y)$.

Известно, что DES имеет 4 слабых ключа

00000001 × 8

11111110 × 8

11100000 × 4 | 11110001 × 4

00011111 × 4 | 00001110 × 4

В случае слабого ключа возможно частичное дешифрование при наличии достаточно длинных открытого текста и шифртекста.

3. Полуслабые ключи DES.

Определение 2. Ключ K называется полуслабым ключом, если существует $K_1=K$ такой, что

$\mathrm{DES}_{K_1}(\bullet)=\mathrm{DES}^{-1}{}_K(\bullet)$.

DES имеет не менее 12 полуслабых ключей.

Опасность состоит в том, что преобразования, реализуемые DES для различных ключей, могут образовать группу. Тогда при суперпозиции преобразований для K_1 и K_2 может существовать ключ K_3 такой, что

$\mathrm{D}_{K1}(\mathrm{D}_{K2}(\bullet))=\mathrm{D}_{K3}(\bullet)$.

В этом случае усиления DES при тройном DES не произойдет.

Замечание 2. В 1992 году Wernsdorf доказал, что множество $\{D_K, k\in V_{64}\}$ не образует группу и нет подгрупп.

4. Зависимость знаков шифртекста от знаков открытого текста.

Доказано, что в DES при выбранных перестановках это свойство достигается на 4 циклах (т.е. через 4 цикла каждый бит в выходном блоке зависит от каждого бита во входном блоке).

3.5. Алгоритм IDEA.

Рассмотрим другие способы реализации свойства D в блочных шифрах. В алгоритме IDEA [Massey 94] свойство D достигается использованием следующей схемы.

M-A-бокс IDEA:

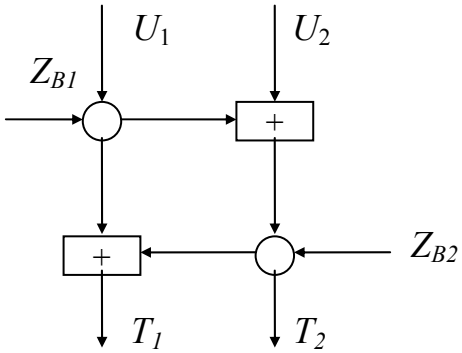


Рис. 1. M-A-бокс IDEA.

Здесь

○ - умножение в мультипликативной группе поля $Z^*_{2^{16}+1} = GF(2^{16}+1)^*$, где $2^{16}+1$ - простое множество для $n = 1,2,4,8,16$;

⊞ - сложение по mod 2^{16}.

Здесь каждый выход зависит от всех четырех входов.

Для реализации свойства *C* в IDEA используется следующая схема

$Y=X \otimes Z_A$, т.е. $(Y_1,Y_2,Y_3,Y_4)=(X_1 \boxplus Z_{A1}, X_2 \boxplus Z_{A2}, X_3 \boxplus Z_{A3}, X_4 \boxplus Z_{A4})$

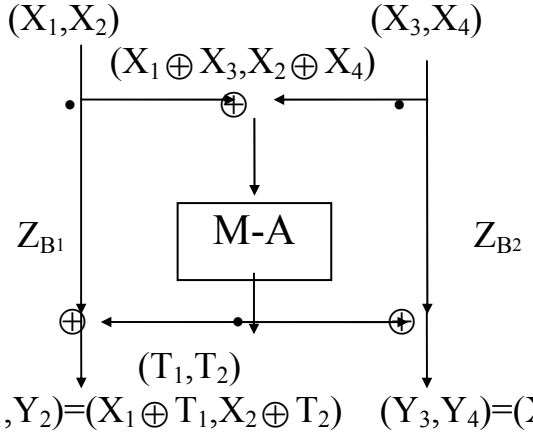
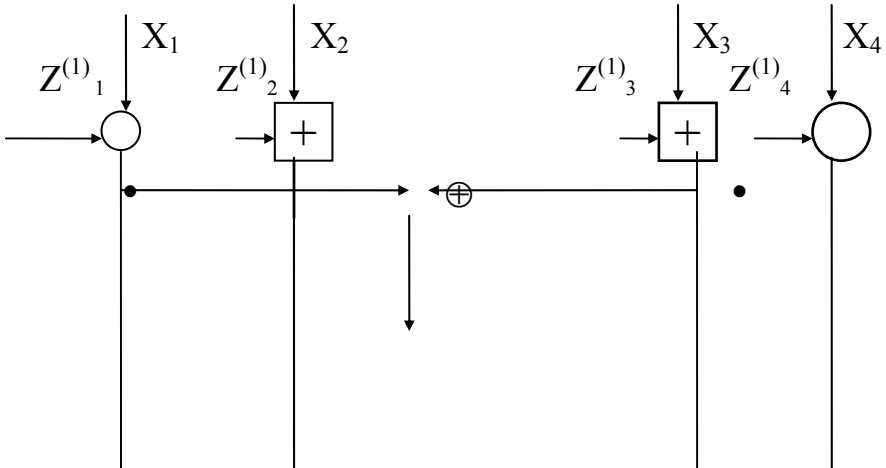


Рис. 2.

Алгоритм IDEA - итеративный блочный алгоритм, в котором описанные преобразования используются следующим образом.



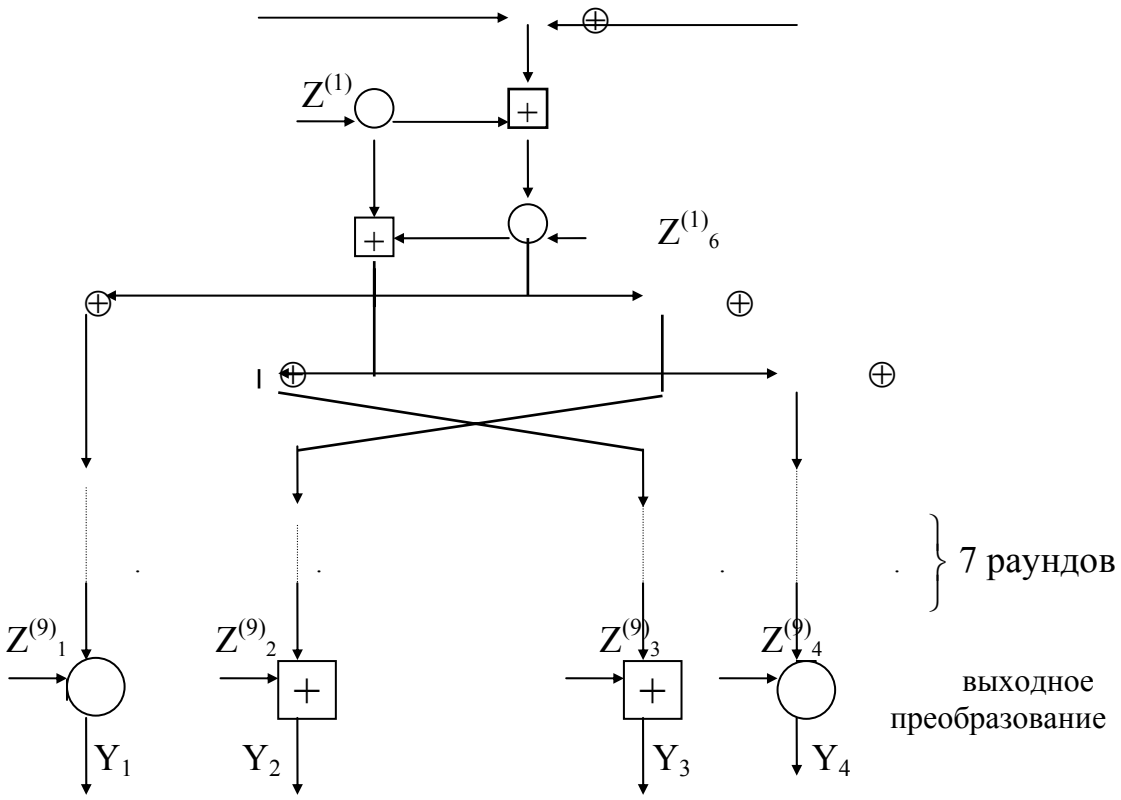


Рис. 3.

При расшифровании используют алгоритм шифрования с изменением порядка использования ключей.

Глава 4. Системы с открытым ключом.

4.1. Алгоритм Евклида и его сложность.

Рассмотрим математические вопросы, связанные с шифрами, использующие операции с целыми числами. Обозначим $\boldsymbol{N}$ – множество натуральных чисел, $\boldsymbol{N}_0$ - множество натуральных чисел с 0, $\boldsymbol{Z}$ –множество целых чисел. Если $a,b \in \boldsymbol{Z}$, $b \neq 0$, то по теореме Евклида существует единственная пара целых чисел q, r таких, что

$$a = bq + r,\ 0 \le r < |b|. \qquad (1)$$

Здесь r называется остатком. Существует несколько обозначений для остатков:

$r = R_b(a)$, $r = a(mod\ b)$ и другие.

Простейшие свойства остатков.

1. $R_{-b}(a) = R_b(a)$, так как

$$\left.\begin{array}{l} a = q \cdot b + r \\ 0 \le r < |b| \end{array}\right\} \Leftrightarrow \begin{cases} a = (-q)(-b) + r \\ 0 \le r < |b| \end{cases}$$

2. $R_b(a + ib) = R_b(a)$, $\forall i \in \boldsymbol{Z}$,

так как $a + ib = (q + i)b + r$.

Если $r = 0$, то b делит a (обозначается b/a).

Из свойства 1 следует, что, если a делится на d, то a делится на $–d$. Поэтому среди всех общих делителей a и b существует наибольший натуральный делитель (обозначается НОД(a, b)). Если НОД(a, b) = 1, то a и b взаимно просты.

Лемма 1. *Для любых a, b, i* $\in \boldsymbol{Z}$

НОД(a, b) = НОД(a+ ib, b).

Доказательство. Если d/a, d/b, то $a = Q_1 d$, $b = Q_2 d$. Тогда $a + ib = d(Q_1 + iQ_2)$, то есть $d/(a + ib)$. Значит НОД(a, b) ≤ НОД(a+ ib, b). Аналогично, пусть $d/(a + ib)$ и d/b, то есть $a + ib = Q_1 d$, $b = Q_2 d$. Тогда $a = d(Q_1 - iQ_2)$, то есть d/a. Это значит, что НОД(a, b) ≥ НОД(a+ ib, b). Из полученных неравенств следует, что НОД(a, b) = НОД(a+ ib, b). Лемма доказана.

Равенство в следующей лемме называется рекурсией Евклида.

Лемма 2. *Для любых* n_1, $n_2 \in \boldsymbol{Z}$, $n_2 \neq 0$,

$$НОД(n_1, n_2) = НОД(n_2, R_{n_2}(n_1)).$$

Доказательство. Так как $n_1 = n_2 q + R_{n_2}(n_1)$, то по лемме 1:

НОД(n_1, n_2) = НОД(n_1-$q\,n_2$, n_2) = ($R_{n_2}(n_1)$, n_2).

Лемма доказана.

Следствие. Если НОД(n_1,n_2) = НОД($R_{n_2}(n_1)$,n_2) = = НОД($R_{R_{n_2}(n_1)}$, $R_{n_2}(n_1)$= …=НОД(0, r),

то r = НОД(n_1, n_2). (2)

Данный способ нахождения НОД(n_1,n_2) называется алгоритмом Евклида.

Пример 1.

НОД(54,30) = НОД(24,30) = НОД(6,24) = НОД(0,6) = 6,

НОД(156,117) = НОД(39,117) = НОД(0,39) = 39.

В криптографии возможность применения любого алгоритма определяется сложностью этого алгоритма. Оценим сложность алгоритма Евклида. Будем считать за одну операцию одно деление в алгоритме Евклида. Отметим, что каждое равенство в (2) связано с одной операцией деления. Обозначим через $m(d)$ минимальное число $n_1 > 0$, для которого существует n_2, $n_1 > n_2 > 0$ такое, что НОД(n_1, n_2) вычисляется при помощи (2) за d операций деления.

Легко вычислить $m(d)$ для малых значений d.

$m(1) = 2$ ($n_1 = 2, n_2 = 1$);

$m(2) = 3$ ($n_1 = 3, n_2 = 2$);

$m(3) = 5$ ($n_1 = 5, n_2 = 3$).

Пусть $n_1 = m(d)$ и n_2 - такое, что НОД(n_1, n_2) вычисляется за d шагов алгоритма Евклида (2). Обозначим $n_1 = q_1 n_2 + r$, $n_2 = q_2 r + r_2$. Тогда для вычисления НОД(n_2, r) надо (d –1) операции в алгоритме Евклида, а для вычисления НОД(r, r_2) надо (d –2) операции в алгоритме Евклида. Из определения n_1 следует

$m(d) = q_1 n_2 + r$. (3)

Вместе с тем, $n_2 \geq m(d\text{-}1)$. Это следует из того, что для n_2 существует r, позволяющее вычислять НОД(n_2, r) за (d –1) делений, а $m(d\text{-}1)$ – наименьшее из натуральных чисел, обладающих этим свойством. Аналогично, для нахождения НОД(r, r_2) требуется (d –2) деления, а $m(d\text{-}2)$ – наименьшее из таких чисел. Тогда $r \geq m(d\text{-}2)$. Из этих оценок и (3) следует, что

$m(d) \geq m(d\text{-}1) + m(d\text{-}2)$. (4)

Рассмотрим рекуррентное соотношение

$\Phi(d) = \Phi(d\text{-}1) + \Phi(d\text{-}2)$. (5)

При начальных условиях $\Phi(1) = 2$, $\Phi(2) = 3$. Тогда из неравенства (4) и равенства (5) и начальных значений $\Phi(1) = m(1)$, $\Phi(2) = m(2)$ следует, что $\Phi(d) \leq m(d)$ для всех $d \geq 1$. Рекуррентное соотношение (5) определяет

числа Фибоначчи (1202 г.). Найдем эти числа методом производящих функций. Пусть $\sum_{d=1}^{\infty} \Phi(d)\, x^d = \Phi(x)$ – формальный ряд. Для $d \geq 3$

$$\Phi(d)\, x^d = x\Phi(d\text{-}1)x^{d-1} + x^2\, \Phi(d\text{-}2)x^{d-2}.$$

Суммируем по допустимым d.

$$\sum_{d=3}^{\infty} \Phi(d)\, x^d = x \sum_{d=3}^{\infty} \Phi(d\text{-}1)x^{d-1} + x^2 \sum_{d=3}^{\infty} \Phi(d\text{-}2)x^{d-2}.$$

Отсюда

$$\Phi(x) - 3x^2\text{-}2x = x(\Phi(x) - 2x) + x^2\ \Phi(x).$$

$$\Phi(x)(1\ \text{-}\ x\text{-}\ x^2) = x^2 + 2x.$$

$$\Phi(x) = \frac{x^2+2x}{1-x-x^2} = \text{-}1\ \text{-}\ \frac{x+1}{x^2+x-1} = \text{-}1\ \text{-}\ \frac{x+1}{(x+\frac{1+\sqrt{5}}{2})(x+\frac{1-\sqrt{5}}{2})} =$$

$$= \text{-}1\ \text{-}\ \frac{(\frac{\sqrt{5}-1}{2})}{\sqrt{5}(x+\frac{1+\sqrt{5}}{2})}\ \text{-}\ \frac{(\frac{1+\sqrt{5}}{2})}{\sqrt{5}(x+\frac{1-\sqrt{5}}{2})}.$$

$$\Phi(d) = \left(\frac{5+3\sqrt{5}}{10}\right)\left(\frac{1+\sqrt{5}}{2}\right)^d + \left(\frac{5-3\sqrt{5}}{10}\right)\left(\frac{1-\sqrt{5}}{2}\right)^d \approx (1{,}171)(1{,}618)^d.$$

Отсюда

$$d \cong 1{,}44 \log_2 \Phi(d) - 0{,}328.$$

Тогда из неравенства $\Phi(d) \leq m(d)$ следует

$$d \leq 1{,}44 \log_2 m(d).$$

Следовательно, для всех n

$$d \leq 1{,}44 \log_2 n_1$$

определяет верхнюю границу сложности алгоритма Евклида.

Пример 2.

$$n \approx 10^{100} \Rightarrow d \leq 1{,}44 \log_2 10^{100} \approx 478.$$

Существуют другие алгоритмы вычисления НОД. Алгоритм Стейна [Massey 94] основан на следующих равенствах:

1. если n_1 и n_2-чётные, то

$$\text{НОД}(n_1, n_2) = 2\,\text{НОД}(\frac{n_1}{2}, \frac{n_2}{2});$$

2. если n_1-чётное, n_2-нечётное, то

$$\text{НОД}(n_1, n_2) = \text{НОД}(\frac{n_1}{2}, n_2);$$

3. если n_1 и n_2-нечётные, то

$$\text{НОД}(n_1, n_2) = \text{НОД}(\frac{n_1 - n_2}{2}, n_2).$$

Сложность алгоритма Стейна. Пусть $\mu(s)$ – минимальное значение $n_1 + n_2$, что s делений потребуется для нахождения НОД(n_1, n_2), $n_1 \geq n_2 > 0$. Тогда

$$s \leq \log_2 n_1 + 1. \qquad (6)$$

Пример 3.

μ(1)=2 (n_1=1, n_2=1);

μ(2)=4 (n_1=3, n_2=1).

4.2. Арифметика остатков

Определение 1. a сравнимо с b по модулю n

$$a \equiv b(n), \qquad (1)$$

если $R_n(a) = R_n(b)$.

Отношение (1) есть отношение эквивалентности. Это отношение разбивает Z на непересекающиеся классы вычетов. Класс вычетов будем обозначать $[a]$, $a, b \in [a] \Leftrightarrow a \equiv b(n)$.

$Z / n = \{[a]\}$ -коммутативное кольцо:

1. $[a] + [b] = [a + b]$ корректно определенная коммутативная, ассоциативная операция.

2. $[a] \cdot [b] = [a \cdot b]$ корректно определенная коммутативная, ассоциативная операция.

3. $([a] + [b]) \cdot [c] = [a \cdot c] + [b \cdot c]$.

Докажем, что результат операции сложения не зависит от выбора представителей (корректность).

$a + b = nq_1 + r_1 + nq_2 + r_2, r_1 + r_2 = n\tilde{q} + r,$

$a' + b' = nq_1^1 + r_1 + nq_2^1 + r_2, r_1 + r_2 = n\tilde{q} + r.$

Для операции умножения корректность доказывается аналогично.

Лемма 1.

$a \in [b] \Leftrightarrow a - b = kn$.

Доказательство непосредственно следует из (1).

Лемма 2. В любом классе вычетов существует единственное наименьшее неотрицательное число.

Доказательство.

$\forall a \in N, \quad a \in [a] \Rightarrow \exists r < n \Rightarrow r \in [a].$

Если $a > n$: $a = qn + r$, $r < n$, то $a - r = qn$ и по лемме 1 $r \in [a]$.

Докажем единственность от противного. Предположим, что $\exists\ r_1 \neq r$, $r_1 < n$, $r_1 \in [a]$. Пусть $r_1 < r$, тогда $r - r_1 = qn > 0 \Rightarrow q \geq 1$, но так как $r < n$, $r_1 < n$, то $r - r_1 < n$ – противоречие. Следовательно, не существует такого r_1. Лемма доказана.

С другой стороны, $\forall\ a,b \in \mathrm{N}, \quad a > b > 0, \quad R_b(a) = r = a(\mathrm{mod}\, b)$.

Следующие свойства остатков важны для дальнейшего.

1. $(c * d)(\mathrm{mod}b) = (c(\mathrm{mod}b) * d(\mathrm{mod}b))(\mathrm{mod}b)$,

где $* \in \{+, -, \times\}$.

2. $(a\times(c+d))(\mathrm{mod}b) = ((a\times c)(\mathrm{mod}b) + (a\times d)(\mathrm{mod}b))(\mathrm{mod}b)$.

Использование вычетов в криптографии основано на существовании алгоритма быстрого возведения в степень.

Пусть

$$a^{2\cdot 2}(\mathrm{mod}\, n) = (a^2)^2(\mathrm{mod}\, n) = (a^2(\mathrm{mod}\, n))^2(\mathrm{mod}\, n),$$

где $a^2(\mathrm{mod}\ n)$ – остаток от деления a^2 на n, следовательно, его проще возводить в квадрат, чем число a^2. Таким образом, любая степень числа два позволяет быстрее возводить в степень, если последовательно приводить промежуточные результаты по модулю n.

$$a^{2^k} \mod n = ((a^2(\mathrm{mod}\, n))^2(\mathrm{mod}\, n))^2(\mathrm{mod}\, n)\ldots$$

Если x – произвольное число, то существует представление

$$x = a_0 \cdot 2^0 + a_1 \cdot 2^1 + \ldots + a_k \cdot 2^k$$

и тогда возведение в степень опирается на быстрый алгоритм возведения в степень числа 2:

$$a^x(\mathrm{mod}\, n) = (a^{a_0\cdot 2^0}(\mathrm{mod}\, n) \cdot a^{a_1\cdot 2^1}(\mathrm{mod}\, n)\ldots a^{a_k\cdot 2^k}(\mathrm{mod}\, n))\,(\mathrm{mod}\, n)$$

Пример 1.

$$3^{12}(\mathrm{mod}19) = (3^8 \mathrm{mod}19)(3^4 \mathrm{mod}19)\,(\mathrm{mod}19);$$

$$3^4(\mathrm{mod}19) = (3^2(\mathrm{mod}19))^2\,(\mathrm{mod}19) = 9^2\,(\mathrm{mod}19) = 5;$$

$$3^8(\mathrm{mod}19) = (3^4(\mathrm{mod}19))^2\,(\mathrm{mod}19) = 5^2(\mathrm{mod}19) = 6;$$

$$3^{12}(\mathrm{mod}19) = (6\cdot 5)(\mathrm{mod}19) = 11.$$

Можно доказать, что возведение в степень k требует в среднем $1{,}5k$ операций.

4.3. Основные теоремы о вычетах.

Пусть n взаимно-простое с a число, $a \neq 0$; $r_0=0$, $r_1,\ldots,r_{n-1}$ - наименьшие неотрицательные представители различных классов вычетов mod n. Рассмотрим множество чисел:

$u_0 = (ar_0)(\mathrm{mod}n)$, $u_1 = (ar_1)(\mathrm{mod}n),\ldots, u_{n-1} = (ar_{n-1})(\mathrm{mod}n)$.

Лемма 1. Числа $u_0, u_1,\ldots, u_{n-1}$ - различные.

Доказательство. Предположим противное. Тогда $\exists\ i > j$: $u_i = u_j$ и по лемме 1 п. 4.2 $ar_i - ar_j = nq$, $q \in Z$. Тогда

$a(r_i - r_j) = nq$.

Так как n простое и взаимно простое с a, то n делит $(r_i - r_j)$. Однако по лемме 1 п. 4.2 это противоречит тому, что r_i и r_j выбирались из разных классов вычетов. Лемма доказана.

Следствие 1. Числа $u_0, u_1,\ldots, u_{n-1}$ - наименьшие неотрицательные представители всех классов вычетов.

Следствие 2. Среди чисел $u_0, u_1,\ldots, u_{n-1}$ есть 1.

Следствие 3. Если n простое, НОД(a,n) = 1, то уравнение $ax \equiv 1(\mathrm{mod}n)$ имеет единственное решение с точностью до класса.

Доказательство следует из следствия 2.

Определение1. Функция Эйлера $\varphi(n)$, $n \in N$, определяется как число натуральных чисел, меньших n и взаимно простых с n, то есть

$\varphi(n) = |\{a: \text{НОД}(a,n) = 1, a<n\}|$.

Если n простое, то $\varphi(n) = n-1$. Если $n=pq$, где p и q простые, то $\varphi(n)=(p-1)(q-1)$.

Пусть для $n \in N$ $r_1, r_2,\ldots, r_{\varphi(n)}$ - все взаимно простые с n натуральные числа, меньшие n. Будем называть этот набор системой представителей для n. Пусть $a \in Z$ и НОД(a,n) = 1. Рассмотрим набор чисел

$u_1 = (ar_1)(\mathrm{mod}n), u_2 = (ar_2)(\mathrm{mod}n),\ldots, u_{\varphi(n)} = (ar_{\varphi(n)})(\mathrm{mod}n)$. (2)

Лемма 2. Если НОД(a,n) = 1, то набор чисел $u_1, u_2,\ldots, u_{\varphi(n)}$ является системой представителей для n.

Доказательство.

Сначала докажем, что $[u_1], [u_2],\ldots, [u_{\varphi(n)}]$ -различны. Предположим противное. Тогда существуют u_i и u_j такие, что

$u_i \equiv u_j(\mathrm{mod}n)$.

Отсюда следует $ar_i - ar_j = nq$ при некотором q. Так как НОД(a,n) = 1, то в произведении $a(r_i - r_j)$ сомножитель $(r_i - r_j)$ делится на n.Это значит, что r_i

и r_j лежат в одном классе вычетов по mod$n$. Пусть $r_i > r_j$, тогда $r_i - r_j = n$. Это противоречит тому, что $0 < r_i - r_j < n$. Каждое из чисел $u_1, u_2, \ldots, u_{\varphi(n)}$ меньше n, они различные и их $\varphi(n)$ штук. Докажем, что каждое из чисел u_i взаимно просто с n. Числа ar_i - взаимно просты с n, так как

НОД$(a,n) = 1$, НОД$(n,r_i) = 1$, i=1,…,$\varphi(n)$. (3)

Тогда, если u_i имеет с n общий делитель $d>1$, то из равенства $ar_i = nq + u_i$ следует, что $d | ar_i$. Это противоречит (3). Отсюда следует, что $u_1, u_2, \ldots, u_{\varphi(n)}$ - система представителей. Лемма 2 доказана.

Лемма 2 является основой для доказательства ряда важных для криптографических приложений результатов.

Теорема 1. *Если $n \in N$, $a \in Z$, НОД$(a,n) = 1$, то уравнение $ax \equiv 1 (\mathrm{mod} n)$ имеет единственное решение в кольце вычетов по mod$n$.*

Доказательство. Так как 1 принадлежит системе представителей для n, то в наборе $u_1, u_2, \ldots, u_{\varphi(n)}$ из леммы 2 найдется 1. Соответствующее r_i такое, что

$1 = u_i = ar_i (\mathrm{mod} n)$

определяет искомое . Теорема доказана.

Теорема 2 (*Обобщенная теорема Евклида*). *Для любых a и b из Z найдутся u, v из Z, такие, что*

$au + bv =$ НОД(a,b).

Доказательство. Пусть сначала НОД$(a,b) = 1$, тогда по теореме 1 уравнение $ax \equiv 1 (\mathrm{mod} b)$ имеет единственное решение $x \equiv u (\mathrm{mod} b)$ в кольце вычетов по модулю b. Это означает, что при некотором v:

$au - 1 = bv$.

Если НОД$(a,b) = d$, то $a = a_1 d$, $b = b_1 d$ и НОД$(a_1, b_1) = 1$. Тогда для a_1 и b_1 найдутся u и v, что $a_1 u + b_1 v = 1$. Умножая обе части равенства на d, получим требуемое равенство. Теорема доказана.

Теорема 3 (*Теорема Эйлера*). *Если $n \in N$, $a \in Z$, НОД$(a,n) = 1$, то*

$a^{\varphi(n)} (\mathrm{mod} n) = 1$.

Доказательство.

Из леммы 2 следует, что если $r_1, r_2, \ldots, r_{\varphi(n)}$ - система представителей для n, то

$u_1 = (ar_1)(\mathrm{mod} n), u_2 = (ar_2)(\mathrm{mod} n), \ldots, u_{\varphi(n)} = (ar_{\varphi(n)})(\mathrm{mod} n)$.

также система представителей для n. Тогда

$u_1 u_2 \ldots u_{\varphi(n)} \equiv r_1 r_2 \ldots r_{\varphi(n)} (\mathrm{mod} n)$

и

$$u_1 u_2 \dots u_{\varphi(n)} \equiv a r_1 a r_2 \dots a\, r_{\varphi(n)} (\mathrm{mod} n).$$

Из этих равенств при некотором q имеем

$$r_1 r_2 \dots r_{\varphi(n)} = a^{\varphi(n)} r_1 r_2 \dots r_{\varphi(n)} + nq. \qquad (4)$$

Так как НОД(r_i,n)=1, i=1,…, $\varphi(n)$, то НОД($r_1 r_2 \dots r_{\varphi(n)}$,$n$)=1. Поэтому в равенстве (4) q делится на $r_1 r_2 \dots r_{\varphi(n)}$. Отсюда получаем

$$1 = a^{\varphi(n)} + nq'.$$

Полученное равенство эквивалентно утверждению теоремы.

Следствие 4 (Малая теорема Ферма). Если n - простое число, НОД(a,n)= 1, то

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

Следствие 5. Если НОД(a,n) = 1, то

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

То есть, при известном значении функции $\varphi(n)$ обращение в кольце вычетов сводится к возведению в степень.

В заключение приведем без доказательства полезную теорему [Massey 94].

Теорема 4. (*Китайская теорема об остатках*).

Пусть $m_1, \dots, m_k \in Z, \quad НОД(m_i, m_j) = 1, \quad i \neq j.$ *Тогда для* $\forall\, r_1, \dots, r_k \in Z, \quad 0 \le r_i < m_i, \quad i = \overline{1,k}$ $\exists$ *единственное решение* $0 \le x < n = m_1 \dots m_k$ *системы* $R_{m_i}(x) = r_i \quad i = 1, \dots, k$.

4.4. Квадратичные вычеты.

Пусть p- простое, $a < p$, $p > 2$.

Определение 1. a – квадратичный вычет $\Leftrightarrow \exists\, x: \quad x^2 \equiv a(p)$.

Пример 1. Пусть $p = 7$, тогда 1, 2, 4 – квадратичные вычеты, а 3, 5, 6 – не квадратичные вычеты:

$$\left.\begin{array}{l} 1^2 = 1 \equiv 1 \,(\mathrm{mod}\, 7) \\ 2^2 = 4 \equiv 4 \,(\mathrm{mod}\, 7) \\ 3^2 = 9 \equiv 2 \,(\mathrm{mod}\, 7) \\ 4^2 = 16 \equiv 2 \,(\mathrm{mod}\, 7) \\ 5^2 = 25 \equiv 4 \,(\mathrm{mod}\, 7) \\ 6^2 = 36 \equiv 1 \,(\mathrm{mod}\, 7) \end{array}\right\} \quad \nexists\, x: \quad x^2 = 3, 5, 6 \ (\mathrm{mod}\ 7)$$

Теорема 1. *Существует* $\frac{p-1}{2}$ *квадратичных вычетов и* $\frac{p-1}{2}$ *не квадратичных вычетов в GF(p). У каждого квадратичного вычета существуют два корня* α_1 и α_2 *такие, что* $0 < \alpha_1 \le \frac{p-1}{2}$, $\frac{p-1}{2} < \alpha_2 \le p-1$, *где* α_1 *называется основным корнем.*

Доказательство. $\beta \equiv \alpha^2 (p) \Rightarrow \beta \equiv (-\alpha)^2 (p)$. При $p > 2$ $[\alpha] \neq [-\alpha]$, следовательно, уравнение $\beta = x^2 (\mathrm{mod} p)$ имеет не менее двух корней. Если $\beta = \alpha^2 (\mathrm{mod} p)$ и $\beta = \gamma^2 (\mathrm{mod} p)$, то

$\alpha^2 - \gamma^2 \equiv 0(p)$,

$(\alpha - \gamma)(\alpha + \gamma) \equiv 0(p)$.

Тогда или $\alpha \equiv -\gamma(n)$ или $\alpha \equiv \gamma(n)$. Отсюда β имеет ровно два значения корня квадратного или ни одного. Ровно половина ненулевых элементов GF(p) есть квадратичные вычеты.

Теорема 2. *Если n = p · q, где p, q – простые, то* $\exists$ $\frac{(p-1)(q-1)}{4}$ *квадратичных вычетов по modn и у каждого квадратичного вычета 4 корня.*

Пример 2. $n = 35 = 5 \cdot 7 \Rightarrow \exists \frac{4 \cdot 6}{4} = 6$ квадратичных вычетов по mod 35: 1, 4, 9, 11, 16, 29. Каждый квадратичный вычет имеет ровно 4 корня.

Определение 2. Для целого *a* и простого *p*>2 символ Лежандра определяется следующим образом:

L(*a, p*) = 0, если *p/a;*

L(*a, p*) = 1, если *a*- квадратичный вычет по mod *p;*

L(*a, p*) = -1, если *a*- не квадратичный вычет по mod *p*.

Теорема 3. $L(a, p) = a^{\frac{p-1}{2}} (\mathrm{mod}\, p)$.

Доказательство.

1. $p / a \Rightarrow (p \cdot a)^{\frac{p-1}{2}} (\mathrm{mod}\, p) = 0$.

2. *a*- квадратичный вычет, то есть $x^2 = a (\mathrm{mod}\, p)$. Тогда по теореме Ферма

$$a^{\frac{p-1}{2}} (\mathrm{mod}\, p) = x^{2 \cdot \frac{p-1}{2}} (\mathrm{mod}\, p) = x^{p-1} \equiv 1 (\mathrm{mod}\, p).$$

Таких a ровно $\frac{p-1}{2}$. Рассмотрим уравнение

$$x^{p-1} \equiv 1 \pmod{p}.$$

Любое a – его корень. Отсюда каждое a является корнем одного из сомножителей в следующем уравнении:

$$(x^{\frac{p-1}{2}} - 1)(x^{\frac{p-1}{2}} + 1) \equiv 0 \pmod{p}.$$

Значит, $\frac{p-1}{2}$ значений квадратичных вычетов a являются корнями уравнения

$$x^{\frac{p-1}{2}} - 1 \equiv 0 \pmod{p}.$$

Тогда по основной теореме алгебры остальные a, не являющиеся квадратичными вычетами, являются корнями уравнения

$$x^{\frac{p-1}{2}} + 1 \equiv 0 \pmod{p}.$$

Если a- не квадратичный вычет, тогда

$$a^{\frac{p-1}{2}} = -1 \pmod{p}.$$

Теорема доказана.

Определение 3. Пусть p- простое, $g < p$. g называется примитивным элементом мультипликативной группы поля GF(p) (или примитивным элементом по mod p), если $\forall\ 0 < n \le p\text{-}1 \quad \exists\ a:\ n \equiv g^a \pmod{p}$.

Пример 3. При p = 11, 2 есть примитивный мультипликативной группы GF(11).

$2^{10} = 1024 \equiv 1 \,(\mathrm{mod}11);$

$2^{1} = 2 \equiv 2(\mathrm{mod}11);$

$2^{8} = 256 \equiv 3(\mathrm{mod}11);$

$2^{2} = 4 \equiv 4(\mathrm{mod}11);$

$2^{4} = 16 \equiv 5(\mathrm{mod}11);$

$2^{9} = 519 \equiv 6(\mathrm{mod}11);$

$2^{7} = 128 \equiv 7(\mathrm{mod}11);$

$2^{3} = 8 \equiv 8(\mathrm{mod}11);$

$2^{6} = 64 \equiv 9(\mathrm{mod}11);$

$2^{5} = 32 \equiv 10(\mathrm{mod}11).$

Каждое число $n \in (1;\ 10)$ может быть представлено в виде $2^{a} \,(\mathrm{mod}\ p)$.

Для p = 11 числа 2, 6,7,8 являются примитивными элементами мультипликативной группы GF(11). Остальные не являются примитивными элементами.

Замечание. В GF(p) $\varphi(p\text{-}1)$ примитивных элементов.

Нахождение примитивного элемента, вообще говоря, не является простой задачей. Примитивный элемент находится легко, если известна факторизация p-1. Пусть $g_1, \ldots, g_n$ - простые множители p-1. Для того чтобы проверить, является ли g примитивным элементом по mod p, вычисляется $l = g^{\frac{(p-1)}{g_i}} \,(\mathrm{mod}\ p), \quad i = \overline{1,n}$. Если для некоторого i это число равно 1, то g-не примитивный элемент. Если это не выполняется ни для какого $i = \overline{1,n}$, то g- примитивный элемент.

Пример 4. При p = 11 имеем p - 1 = 10 = 2 · 5. Проверим, что число 2 является примитивным элементом мультипликативной группы GF(11).

$2^{\frac{(11-1)}{5}} = 4(\mathrm{mod}11) \neq 1,$

$2^{\frac{(11-1)}{2}} = 10(\mathrm{mod}11) \neq 1,$

следовательно, 2 – примитивный элемент. Проверим 3.

$$3^{\frac{(11-1)}{5}} = 9(\text{mod}11) \neq 1$$

$$3^{\frac{(11-1)}{2}} = 243(\text{mod}11) = 1,$$

следовательно, 3 не примитивный элемент мультипликативной группы GF(11).

4.5. Факторизация. Логарифмирование в конечных полях.

Факторизация числа означает нахождение его простых множителей.

Пример 1.

$10 = 2 \times 5,$

$60 = 2 \times 2 \times 3 \times 5,$

$252601 = 41 \times 61 \times 101,$

$2^{113} - 1 = 3391 \times 23279 \times 65933 \times 1868569 \times 1066818131868207.$

Проблема факторизации – одна из старейших в теории чисел. Сложность самого быстрого алгоритма факторизации [Schn96]:

$$e^{\sqrt{\ln n}\cdot\sqrt{\ln\ln n}}$$

Пусть $\pi(n)$ – число простых чисел $\leq n$.

Теорема. (Чебышева о числе простых чисел [Massey 94]).

$$\lim_{n\to\infty} \frac{\pi(n)}{\frac{n}{\ln n}} = 1.$$

Все современные тесты на простоту основаны на обобщениях теоремы Ферма.

Если α имеет порядок N ($\alpha^N = 1(\text{mod}p)$) в мультипликативной группе поля GF(p), что предполагает, что $\alpha^0 = 1, \alpha, \alpha^2, \ldots, \alpha^{N-1}$ - различны, и $\beta = \alpha^i$, где $0 \leq i < N$. Тогда i называется дискретным логарифмом от β по основанию α и обозначается $i = \log_\alpha \beta$.

$$y = f(x) = \alpha^x, \qquad 0 \leq x < N,$$

$$\Updownarrow$$

$$x = f^{-1}(y) = \log_\alpha y$$

при этом:

(1) вычислять экспоненту легко: требуется не более $2[\log_2 N]$ операций в алгоритме быстрого возведения в степень;

(2) вычисление дискретного логарифма в GF(p) является очень сложной задачей, когда p-1 имеет большой простой множитель.

Для вычисления $x = \log_a y$ в GF(p), когда α имеет мультипликативный порядок N ($\alpha^N = 1(\mathrm{mod}p)$), используют алгоритм Shank [Massey 94]. Выбирается некоторое целое d, где $d \approx \sqrt{N}$.

Тогда $x = Qd + r,\ 0 \le r < d \approx \sqrt{N}$ и

$$0 \le Q < \frac{N}{\alpha} \approx \sqrt{N}.$$

Далее все вычисления ведутся в GF(p).

1. Для $\rho = 0, 1, \ldots, d\text{-}1$ вычисляем $a^\rho = a \cdot a^{\rho-1}$ и запоминаем $(\alpha^\rho, \rho) \Rightarrow \rho = \log_\alpha(\alpha^\rho)$ - это все логарифмы γ такие, что $\log_\alpha \gamma < d$.
2. Составляем таблицу $(\gamma, \log_\alpha \gamma)$ для $0 \le \log_\alpha \gamma < d$ – таблицу малых логарифмов.
3. Вычисляем $\alpha^{-d} = \alpha^{N-d}$ с помощью алгоритма быстрого возведения в степень.

Заметим, что

$$y = \alpha^x = \alpha^{Qd+r} = a^{Qd} a^r,$$

$$y\alpha^{-Qd} = \alpha^r$$

и, следовательно,

$$y(\alpha^{-d})^Q = a^r.$$

4. Для $q = 0, 1, 2, \ldots$ вычисляем $y \cdot (\alpha^{-d})^q$ до тех пор, пока $y \cdot (\alpha^{-d})^q$ не станет равным некоторому $\gamma = \alpha^\rho$ в таблице.
5. Тогда $x = qd + \rho$ *и* $\alpha^{qd+\rho} = \alpha^{qd} \cdot \alpha^\rho = \alpha^{qd} \cdot y(\alpha^{-d})^q = y$.

Сложность алгоритма имеет порядок $2\sqrt{N}$ операций умножения и память $\sim\sqrt{N}$.

Пример 2. Найдем $x = \log_5 13$, здесь 5- примитивный элемент $GF(23)$, и $\Rightarrow$ $N = 22$ ($5^{22} = 1(\mathrm{mod}23)$).

(0) $d = 5$

(1) $5^0 = 1$

$5^1 = 5$

$\left.\begin{array}{l} 5^2 = 25 = 2 \\ 5^3 = 5 \cdot 2 = 10 \\ 5^4 = 5 \cdot 10 = 4 \end{array}\right\}$ 3 *операции умножения*.

(2)

γ	$\log_\alpha \gamma$
1	0
2	2
4	4
5	1
10	3

(3) $\alpha^{-d} = \alpha^{N-d} = \alpha^{17} = \alpha^{16} \cdot \alpha^{1}$

$$\left.\begin{aligned} &\alpha^1 = 5 \\ &\alpha^2 = 5 \times 5 = 2 \\ &\alpha^4 = 2 \times 2 = 4 \\ &\alpha^8 = 4 \times 4 = 16 \\ &\alpha^{16} = 16 \times 16 = 3 \end{aligned}\right\} \quad \alpha^{-d} = 3 \times 5 = 15$$

(4) $y = 13$ не принадлежит входу в таблице ($q = 0$).

2 операции умножения $\begin{cases} y \cdot \alpha^{-d} = 13 \times 15 = 11 \text{ не принадлежит входу в} \\ \text{таблице } (q = 1), \\ (y \cdot \alpha^{-d}) \cdot \alpha^{-d} = 11 \times 15 = 4 \text{ принадлежит входу} \\ \text{в таблице при } q = 2. \end{cases}$

Следовательно,

$$(4, 4) = (4, \rho) \Rightarrow x = qd + \rho = 2 \cdot 5 + 4 = 14.$$

$$x = \log_5 13 = 14$$

и для его нахождения требуется 5 операций умножения.

Проблема дискретного логарифмирования упрощается, если основание логарифма α имеет порядок N и $N = N_1 \cdot N_2$ и сводится к дискретному логарифмированию по основанию, имеющему порядок N_1, и дискретному логарифмированию по основанию, имеющему порядок N_2 .

Алгоритм [Massey 94] нахождения дискретного логарифма, когда основание логарифма α имеет порядок N и $N = N_1 \cdot N_2$ приведен ниже. Пусть

$$x = \log_\alpha y \quad (a^N = 1, N = N_1 \cdot N_2).$$

(0) Вычисляем $y_1 = y^{N_2}$ с помощью алгоритма быстрого возведения в степень.

(1) Вычисляем $\alpha_1 = \alpha^{N_2}$ с помощью алгоритма быстрого возведения в степень.

(2) Находим $x_1 = \log_{\alpha_1} y_1$.

(3) Вычисляем α^{N-x_1} с помощью алгоритма быстрого возведения в степень.

(4) Вычисляем $y_2 = y \cdot \alpha^{N-x_1}$.

(5) Вычисляем $\alpha_2 = \alpha^{N_1}$ с помощью алгоритма быстрого возведения в степень.

(6) Находим $x_2 = \log_{\alpha_2} y_2$.

(7) Получаем $x = x_2 N_1 + x_1$.

4.6. Схема RSA.

RSA [Schn 96] используется как для шифрования, так и для подписи.

Выберем p и q – большие простые числа. Пусть модуль $n = p \cdot q$, функция $\varphi(n) = (p-1)\cdot(q-1)$ – функция Эйлера. Возьмем произвольное $1 \le e < \varphi(n)$ такое, что НОД(e, $\varphi(n)$) = 1. Тогда существует единственное $1 \le d < \varphi(n)$ такое, что $ed \pmod{\varphi(n)} = 1$.

Система шифрования RSA – это система с открытым ключом, где e –открытый, а d – секретный ключи. Если $0 \le x < n$ – это открытое сообщение, то шифрсообщение получается следующим образом:

$C = x^e \pmod n$.

Возможность расшифрования следует из следующей теоремы.

Теорема 1. *Если p и q – большие простые числа,* $ed \pmod{\varphi(n)} = 1$, *то* $\forall$ x, $0 \le x < n$:

$(x^e)^d \pmod n = x$.

Доказательство. Пусть НОД(x, n) =1. Тогда

$(x^e)^d = x^{e\,d} = x^{k\cdot\varphi(n)+1}$.

Поэтому по теореме Эйлера

$(x^e)^d \pmod n = (x(x^{\varphi(n)} \pmod n)) \pmod n = (x \cdot 1) \pmod n = x$.

Если НОД(x,n) $\neq$ 1, то или $x = 0 \pmod n$, или НОД(x,n) = p, или НОД(x,n) = q. Если $x = 0 \pmod n$, то $x^{e\,d} = 0 \pmod n$. Пусть НОД(x,n) = p. Тогда $x = x_1 p$, где (x_1, n) =1.

$$x^{e\,d} = x^{k(p-1)(q-1)+1} = px_1 p^{k(p-1)(q-1)} x_1^{\,k(p-1)(q-1)} \equiv y \pmod{pq}.$$

Если $mp = y \pmod{pq}$, то $mp = pqk + y$, следовательно, $y = py_1$. Тогда $m \equiv y_1 \pmod q$. Следовательно,

$$x_1((px_1)^{k(p-1)})^{q-1} \equiv y_1 \pmod q.$$

По теореме Ферма $z_1^{\,q-1} \equiv 1 \pmod q$. Поэтому

$x = x_1 p(\mathrm{mod}(pq)) = y\ (\mathrm{mod}n) \equiv x^{e\ d}\ (\mathrm{mod}n)$,

что и требовалось доказать.

Открытый и шифрованный тексты эффективно вычисляются, если известны e и d при помощи алгоритма быстрого возведения в степень. Если искать секретный ключ d по известному открытому ключу e, то надо знать $\varphi(n)$.

Теорема 2. *Вычисление* $\varphi(n) = (p\text{-}1)(q\text{-}1)$ *эквивалентно* (*с точностью до алгоритма полиномиальной сложности*) *факторизации числа* $n = p \cdot q$.

Доказательство. Пусть известны n и $\varphi(n)$. Тогда p и q находятся быстро. Это следует из следующих равенств.

$\varphi(n) = (p\text{-}1)(q\text{-}1) = pq\text{-}p\text{-}q\text{+}1 = n\text{-}p\text{-}q\text{+}1$.

Отсюда

$p+q = n\text{-}\varphi(n)\text{+}1$,

$pq = n$.

По теореме, обратной к теореме Виета, p и q являются корнями квадратного уравнения:

$x^2 - (n - \varphi(n)+1)x + n = 0$.

Вычисление корней – полиномиальный алгоритм. Обратно, если известны p и q, $pq = n$, то $\varphi(n) = (p\text{-}1)(q\text{-}1)$. Теорема доказана.

Подпись RSA.

Пусть M – сообщение, которое надо подписать. Подпись получается по следующему алгоритму:

$C = M^d\ (\mathrm{mod\ n})$,

тогда (M, C) - сообщение с подписью. Проверка подписи осуществляется следующим образом

$C^e\ (\mathrm{mod\ n}) = M^{e\ d}\ (\mathrm{mod\ n}) = M'$.

Если $M = M'$, то подпись верна.

4.7. Атаки на RSA.

1. Активная атака с помощью выбранного шифртекста.

Пусть злоумышленник E перехватывает шифрованное сообщение c от корреспондента A к корреспонденту B. Пусть $c = z^e\ (\mathrm{mod}n)$, где z- открытый текст, e- открытый ключ и $z = c^d\ (\mathrm{mod}\, n)$, где d- секретный ключ. E хочет прочитать открытый текст z. Для этого E выбирает число r, $r < n$, и вычисляет с помощью открытого ключа

$$x = r^e \pmod n,$$

$$y = x \cdot c \pmod n,$$

$$t = r^{-1} \pmod n,$$

так, что $r = x^d \pmod n, \quad t = x^{-d} \pmod n.$

Пусть *E* подписывает у *A* y с помощью секретного ключа *d*, т.е.

$$u = y^d \pmod n.$$

Затем *E* вычисляет

$$t \cdot u \pmod n = x^{-d} \cdot (y^d \pmod n) = x^{-d} \cdot x^d \cdot z^{ed} \pmod n = z.$$

Таким образом, *E* читает *z*.

2. Предположим, что *E* хочет получить подпись на число *N* у *A,* а *A* никогда не подпишет *N*. Тогда *E* выбирает произвольным образом *X* и вычисляет

$$Y = X^e \pmod n.$$

Затем *E* вычисляет $M = Y \cdot N$ и посылает *A* на подпись. *A* подписывает *M* и возвращает *E* число $M = M^d \pmod n$. *E* вычисляет

$$M^d X^{-1} \pmod n = (Y \cdot N)^d \cdot X^{-1} \pmod n =$$

$$= X^{e \cdot d} \cdot N^d \cdot X^{-1} \pmod n = N^d \pmod n,$$

что и является подписью *N*.

3. Пусть *E* хочет подписать у *A* число M_3. Тогда *E* генерирует 2 сообщения M_1 и M_2 так, что

$$M_3 \equiv M_1 \cdot M_2 \pmod n.$$

Подписав у *A* сообщения M_1 и M_2, *E* затем вычисляет подпись для M_3:

$$M_3^d \pmod n = M_1^d \pmod n \cdot M_2^d \pmod n.$$

4. Атака с использованием общего модуля.

Пусть у всех абонентов сети один модуль *n* и различные секретные и открытые ключи. Рассмотрим открытый текст *P* и предположим, что он зашифрован на двух различных открытых ключах e_1 и e_2

$$c_1 = p^{e_1} \pmod n,$$

$$c_2 = p^{e_2} \pmod n, \quad \text{где} \quad \text{НОД}(e_1, e_2) = 1.$$

Тогда в силу обобщённой теоремы Евклида, существуют такие числа *r* и *s*, что

$$r \cdot e_1 + s \cdot e_2 = 1.$$

Пусть *r* отрицательно (отрицательно должно быть либо *r*, либо *s*). Тогда можно вычислить c_1^{-1}, используя алгоритм Евклида, а затем

$$(c_1^{-1})^{-r} \cdot c_2^s = p^{e_1 r + e_2 s} (\mathrm{mod}\, n) = P.$$

5. Атака с малой экспонентой.

Пусть *e* мало, например, 3 и используется для зашифрования одного и того же шифртекста *x* по различным модулям. Соответствующие шифртексты

$$y_1 = x^3 (\mathrm{mod}\, m_1);$$

$$y_2 = x^3 (\mathrm{mod}\, m_2);$$

$$y_3 = x^3 (\mathrm{mod}\, m_3),$$

и пусть m_1, m_2, m_3- взаимно- простые числа. Следовательно, $x^3 < m_1 \cdot m_2 \cdot m_3$. Мы можем найти x^3 из y_1, y_2, y_3, используя обращение китайской теоремы об остатках, а затем найти *x* путём извлечения кубического корня из x^3.

4.8. Криптографические протоколы. Компрометация криптопротоколов.

Рассмотрим протокол связи абонентов сети $A_1, A_2, \ldots, A_N$. с использованием RSA. Пусть у каждого абонента есть справочник, в котором находятся все открытые ключи всех абонентов $e_1, e_2, \ldots, e_N$. Соответственно, $d_1, d_2, \ldots, d_N$ -секретные ключи абонентов.. Абонент A_i хочет послать зашифрованное сообщение абоненту A_j. Пусть x - это открытый текст. Тогда A_i вычисляет

$c = x^{e_j} \pmod{n}$.

Затем

$$A_i \xrightarrow{\quad c \quad} A_j,$$

что означает, что абонент A_i посылает сообщение c абоненту A_j.

Получив сообщение c абонент A_j вычисляет

$x = c^{d_j} \pmod{n}$.

То, что описано выше – это протокол секретной связи в сети. Данный протокол определяется следующими элементами.

1. Участники протокола – абоненты A_i и A_j;
2. Язык протокола – (множество понимаемых участниками слов) – блок длины $\le \log_2 n$ или последовательность блоков (сообщение = слова);
3. Порядок обмена сообщениями. Любой обмен это трехэтапная процедура:

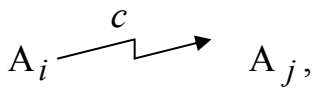
- формирование сообщения у абонента A_i (в нашем случае – это формирование сообщения c);
- передача сообщения абоненту A_j;
- формирование исходных данных для следующего шага протокола или завершения протокола (в нашем случае – это формирование x).

Криптопротокол в отличии от обычного протокола характеризуется наличием у некоторых участников секретов, то есть информации, которая известна одному или группе участников и не должна попасть к не участникам группы (информация может быть измерена информационным потоком или возможностью за приемлемое время вычислить секрет или часть его).

В протоколе связи два секрета: d_j - секрет A_j и x – секрет для A_i и A_j.

Протокол может быть частью другого протокола. Например, рассмотренный ранее протокол секретной связи:

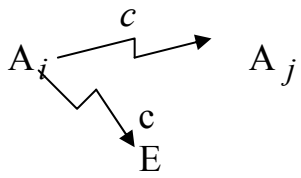
$c = x^{e_j} \pmod{n}$.



$x = c^{d_j} \pmod{n}$ является частью следующего протокола.

Предполагаем расширение числа участников протокола (“пассивный” перехват) за счет добавления участника E, который из канала связи получает сообщение, переданное A_i.

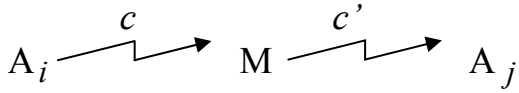
$c = x^{e_j} \pmod{n}$,



$x = c^{d_j} \pmod{n}$.

Так как мы не знаем действий E, то с получением c его участие в протоколе завершено.

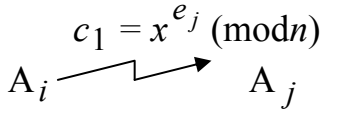
Возможно активное изменение протокола, то есть имеется еще один (активный) участник M:

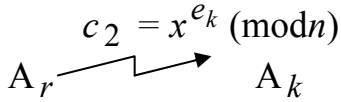


Если M играет роль транслятора или E, то функция протокола выполняется, иначе выполняется другой протокол вместо исходного.

В нашем случае цель модификации протокола – компрометация секрета, то есть получение хотя бы частичной информации о секрете не членами группы.

Пример. Пусть протокол связи включает множество протоколов связи абонентов. Пусть в сети имеется один модуль n. Тогда возможен случай, когда





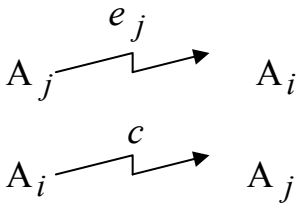
Пусть E перехватывает все сообщения, и предположим, что x для c_1 и c_2 одно и то же. Поскольку E знает e_j и e_k, он может вычислить НОД(e_j, e_k). Если НОД(e_j, e_k)= 1, то в указанных предположениях он

может провести компрометацию протокола секретной связи, опираясь на атку 4 п. 4.7. То есть по обобщенному алгоритму Евклида существуют r и s, что $re_j + se_k = 1$. Если $r < 0$, то Е считает $(c_1)^{-1}$, тогда

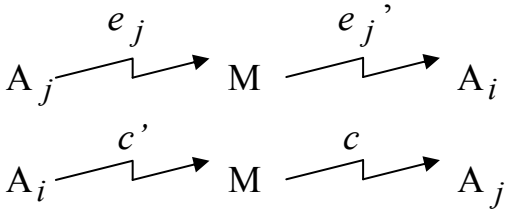
$$(c_1^{-1})^{-r}\ c_2^s = m^{re_j + se_k} (\mathrm{mod} n) = x.$$

Каким образом Е находит одинаковые x по c_1 и c_2 - другая проблема.

Существуют универсальные атаки. Пусть справочника открытых ключей RSA в сети нет. Протокол связи выглядит следующим образом:



Модифицированный протокол, компрометирующий исходный протокол, носит название “человек по середине”.



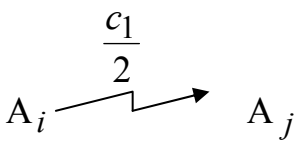
Здесь М передает A_i свой открытый ключ e_j’ вместо полученного от A_i ключа e_j. Тогда

$$c' = x^{e_j'} (\mathrm{mod} n),$$

$$c = x^{e_j} (\mathrm{mod} n).$$

Защита от атаки “человек по середине”. A_i и A_j обмениваются ключами и договариваются о передачи части сообщения. При этом оставшуюся часть посылают только после получения подтверждения о получении первой части сообщения. Такой способ защиты не всегда защищает секрет, но иногда позволяет выявить наличие “человека по середине”. А именно, A_i вычисляет

$$c_1 = x^{e_j} (\mathrm{mod} n)$$



A_j вычисляет

$$c_2 = y^{e_i} (\mathrm{mod} n).$$

$$\mathrm{A}_j \xrightarrow{\frac{c_2}{2}} \mathrm{A}_i$$

Получив сообщения, каждая сторона посылает второй стороне вторые половины сообщений. Если все верно, то сообщения читаются. Если М по середине перехватил $\frac{c_1}{2}$, $\frac{c_2}{2}$, то не может их расшифровать, а для передачи других половинок надо послать что-то абонентам. М вынужден выдумывать, что заведомо создает новый разговор, чем прежде.

Иногда указанный метод не защищает. Пусть A_i и A_j создают ключ k для сеанса по правилу $k = k_i + k_j$. Тогда послав друг другу

$$\mathrm{A}_i \xrightarrow{k_i^{e_j}/2} \mathrm{A}_j$$

$$\mathrm{A}_j \xrightarrow{k_j^{e_i}/2} \mathrm{A}_i$$

по половинке ключа, они попадут к противнику и М, сочинив свои половинки, создает свои ключи k' и k''.

Рассмотрим менее примитивный пример атаки на протокол распределения ключей TMN (Tatebayashi-Matsuzakai-Newman) [Simm 94]. Пусть имеется мобильная сеть. Любая мобильная информация ограничена по вычислительным возможностям и по криптографии. Есть сервер с хорошими вычислительными возможностями и криптографией. Предположим, что сервер безопасен при хранении и использовании своего ключа (а не всех в сети). Любой терминал имеет алгоритм шифрования (DES) и систему с открытым ключом типа RSA: $n = pq$, $c = x^3 \pmod{n}$. Сервер знает p и q и, следовательно, быстро вычисляет $\sqrt[3]{c} \pmod{n}$. Пусть абоненты А и В хотят обменяться ключом k для DES, используя сервер S, которому доверяют. Протокол выглядит следующим образом.

Протокол 1.

1. А вычисляет случайное число r_1 длиной 64 бит. А вычисляет $c_1 = (r_1)^3 \pmod{n}$.

$$\mathrm{A} \xrightarrow{c_1} \mathrm{S}$$

2. S вычисляет r_1 зная p и q. S генерирует γ и

$$\mathrm{S} \xrightarrow{\gamma} \mathrm{B}$$

3. B вычисляет сеансовый ключ для DES r_2. B вычисляет $c_2 = (r_2)^3 \pmod{n}$.

$$B \xrightarrow{c_2} S$$

4. S вычисляет r_2. S вычисляет $E(r_1, r_2) = r_1 \oplus r_2$.

$$S \xrightarrow{E(r_1, r_2)} A$$

5. A вычисляет $E(r_1, r_2) \oplus r_1 = r_2$ - ключ сеанса.

Компрометация протокола.

E подслушивает все, что ему надо. Д – сообщник E, Д и E легальные пользователи сети. Цель E – узнать секрет r_2 (а затем открытый текст). Опираясь на свои возможности, Д и E дополняют протокол 1.

1. E подслушивает $c_2 = (r_2)^3 \pmod{n}$, когда

$$B \xrightarrow{c_2} S$$

2. E выбирает случайное число r (64 бит). Вычисляет $r^3 \pmod{n}$. Вычисляет

$$c_3 = (r_2)^3 r^3 \pmod{n} = (r_2 r)^3 \pmod{n}$$

$$E \xrightarrow{c_3} S$$

с просьбой организовать связь с Д.

3. S вычисляет $\sqrt[3]{r_2^3 r^3} \pmod{n} = r_2 r \pmod{n}$ и

$$S \xrightarrow{\text{вызов}} D$$

4.Д вычисляет $c_4 = (r_k)^3 \pmod{n}$.

$$D \xrightarrow{c_4} S$$

5. S вычисляет r_k. S вычисляет $(r_k + r_2 r) \pmod{n} = c_5$

$$S \xrightarrow{c_5} E$$

6. Так как E и Д в сговоре, то E знает r_k. Тогда E вычисляет из уравнения

$$(r_k + r_2 r) \pmod{n} = c_5$$

$$r_2 = (c_5 - r_k) r^{-1} \pmod{n}.$$

4.9. Система Диффи и Хеллмана.

Система Диффи и Хеллмана реализует протокол открытого распределения ключей [Schn 96].

Пусть абоненты сети A_1, A_2, …, A_N имеют криптоалгоритм $y = T(x, k)$ с секретным ключом. Если каждый абонент имеет секретный ключ для связи с каждым другим абонентом, то таких ключей требуется $\frac{N(N-1)}{2}$. Каким образом можно создать и распределить ключи с учетом развития сети? Пусть имеется защищенный справочник сети. Рассмотрим поле GF(p), p - большое простое число. В GF(p) (p-1) ненулевых элементов, среди них есть примитивные. Пусть α - примитивный элемент и α^0, α^1,…, α^{p-2} образуют всю группу GF(p) по умножению. Пусть x_1, x_2, …, x_N - случайные числа , которые выбираются каждым абонентом A_1, A_2, …, A_N случайно и каждый из них хранит свое число в секрете. Далее каждый абонент из A_1, A_2, …, A_N вычисляет $y_i = a^{x_i} (\mathrm{mod} p)$ и публикует значение y_i в общем справочнике. Тогда секретный ключ для переписки абонентов сети A_i и A_j формируется следующим образом.

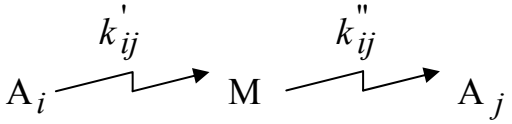
1. A_i берет в общем справочнике число y_j.
2. A_j берет в общем справочнике число y_i.
3. A_i вычисляет $k_{ij} = y_j^{x_i} (\mathrm{mod} p)$.
4. A_j вычисляет $k_{ji} = y_i^{x_j} (\mathrm{mod} p)$.
5. $k = k_{ij} = k_{ji}$ их общий секретный ключ.

В самом деле,

$$k_{ji} = y_i^{x_j} (\mathrm{mod} p) = a^{x_i x_j} (\mathrm{mod} p) = y_j^{x_i} (\mathrm{mod} p) = k_{ij}.$$

Для вычисления этого ключа противнику Е надо уметь решать задачу логарифмирования в конечных полях, а она сложная.

Схема Диффи и Хэлмана также подвержена атаке “человек по середине”.



4.10. Схема шифрования ElGamal.

Схему Диффи и Хэлмана можно модифицировать так, чтобы построить систему шифрования с открытым ключом.

Пусть p – большое простое число, g – примитивный элемент мультипликативной группы GF(p), x случайное число, $x < p$-1. $y = g^x \pmod p$ -открытый ключ, x –секретный ключ. Пусть надо зашифровать сообщение $M<p$.

1. Выбирается случайное число k, взаимно-простое с p - 1.
2. Затем вычисляется

$$a = g^k \pmod p$$

$$b = y^k \cdot M \pmod p$$

Шифртекстом является пара (a, b).

При расшифровании вычисляется a^x и $b/a^x \pmod p$,

$$b/a^x \equiv y^k \cdot M/a^x \equiv g^{k\cdot x}\, M/g^{k\cdot x} \equiv M \pmod p.$$

4.11. Подпись ElGamal.

Для генерации ключевой пары выбираются большое простое число p и примитивный элемент g мультипликативной группы GF(p). Выбирается случайное число x такое, что $x < p$-1. Открытым ключом является $y = g^x \pmod p$; секретным ключом является x.

Схема ElGamal может быть использована для подписи в электронных деньгах и для зашифрования. Стойкость основана на сложности дискретного логарифмирования.

Пусть A должен подписать сообщение M. Выбирается случайное число k, взаимно-простое с p-1: НОД(k; p-1) = 1. Затем вычисляется

$$a = g^k \pmod p.$$

Рассмотрим уравнение

$$M = (x \cdot a + k \cdot b) \pmod{(p - 1)}.$$

По теореме о вычетах $\exists\ k^{-1}$: (M-xa) $k^{-1} \equiv b \pmod{(p\text{-}1)}$. Подписью под M является пара (a, b).

Проверка подписи:

Вычисляем $g^M(p)$ и $y^a \cdot a^b(p)$. Проверяем

$$y^a \cdot a^b \pmod p = g^{a\cdot x} \cdot g^{k\cdot b} \pmod p = g^{a\cdot x + k\cdot b} \pmod p =$$

$$= g^{ax + kk^{-1}(M - xa) + (p-1)nk} \pmod p = g^{M + (p-1)nk} \pmod p = g^M \pmod p.$$

4.12. DSA (DSS).

В основе DSA(DSS) (Digital Signature Algorithm (Digital Signature Standard)) [DSS 94] лежит подпись El Gamal. Пусть p – простое число, q – простое число, такое, что $q|(p\text{-}1)$ и g имеет мультипликативный порядок q, то есть $g^q = 1(\mathrm{mod}p)$.

Лемма. Мультипликативные вычисления степеней g по модулю p эквивалентны арифметическим вычислениям в показателе g по модулю q.

Доказательство.

$$g^x g^y \ (\mathrm{mod}p) = g^{x+y} \ (\mathrm{mod}p) = g^{(x+y)(\mathrm{mod}\,q)+qn} \ (\mathrm{mod}p) =$$
$$= g^{(x+y)(\mathrm{mod}\,q)} g^{qn} \ (\mathrm{mod}p) = \ g^{(x+y)(\mathrm{mod}\,q)} \ (\mathrm{mod}p).$$

Аналогично

$$(g^x)^y (\mathrm{mod}p) = g^{xy(\mathrm{mod}\,q)} \ (\mathrm{mod}p),$$

что и требовалось доказать.

В DSS выбирают $p\sim$ 512 бит, $q\sim$ 160 бит. Пусть x случайное число, $x< p\text{-}1$. Тогда x – секретный ключ, а

$$y = g^x (\mathrm{mod}p) –$$

открытый ключ. Алгоритм подписи сообщения M выглядит следующим образом.

1. Выбирается случайное число $k<q$. Так как НОД(k,q) = 1, то $\exists \ k^{-1}(\mathrm{mod}\,q)$.
2. Вычисляется

$$r = [g^k (\mathrm{mod}p)](\mathrm{mod}q).$$

3. Вычисляется

$$s = \ k^{-1}(M + xr)(\mathrm{mod}\,q).$$

Подписью сообщения M является пара (r,s).

Проверка подписи осуществляется следующим образом. Вычисляется

$$w = \ s^{-1}(\mathrm{mod}\,q).$$

Это можно сделать, так как q простое число.

$$u_1 = (Mw)(\mathrm{mod}q),$$
$$u_2 = (rw)(\mathrm{mod}q),$$
$$v = [(g^{u_1} \ y^{u_2})(\mathrm{mod}p)](\mathrm{mod}q).$$

Если выполняется равенство $v = r$, то подпись подтверждена.

Доказательство.

$$v = [(g^{u_1} \ y^{u_2})(\mathrm{mod}p)](\mathrm{mod}q).$$

Тогда по лемме

$$v = [(g^{M\cdot s^{-1}(\mathrm{mod}\,q)} \ g^{xr\cdot s^{-1}(\mathrm{mod}\,q)})(\mathrm{mod}p)](\mathrm{mod}q) =$$

$= [(g^{\cdot s^{-1}(M+xr)(\mathrm{mod}\,q)})(\mathrm{mod}p)](\mathrm{mod}q) =$

$= [g^{k}(\mathrm{mod}p)](\mathrm{mod}q) = r.$

4.13. ГОСТ 3410.94.

Пусть p – простое число размера 509÷512 бит, q простое число такое, что $q|(p\text{-}1)$. Число $g<p$-1 имеет мультипликативный порядок q, то есть $g^{q} = 1(\mathrm{mod}p)$. p, q, g открыты. Число x, $x<q$, –секретный ключ, открытым ключом является $y = g^{x}(\mathrm{mod}p)$. Алгоритм подписи [ГОСТ 94] сообщения M выглядит следующим образом.

1. Выбирается случайное число k.
2. Вычисляется

$r = [g^{k}(\mathrm{mod}p)](\mathrm{mod}q).$

3. Вычисляется

$s = (Mk + xr)(\mathrm{mod}q).$

Подписью сообщения M является пара (r,s).

Проверка подписи осуществляется следующим образом. Вычисляются

$v = M^{q-2}(\mathrm{mod}q),$

$z_1 = (sv)(\mathrm{mod}q),$

$z_2 = ((q\text{-}r)v)(\mathrm{mod}q),$

$u = [(g^{z_1} y^{z_2})(\mathrm{mod}p)](\mathrm{mod}q).$

Если выполняется равенство $u = r$, то подпись подтверждена.

Доказательство.

$u = [(g^{z_1} y^{z_2})(\mathrm{mod}p)](\mathrm{mod}q) =$

$= [(g^{(xr+kM)M^{q-2}(\mathrm{mod}\,q)+x(q-r)M^{q-2}(\mathrm{mod}\,q)})(\mathrm{mod}p)](\mathrm{mod}q) =$

$= [(g^{xrM^{q-2}(\mathrm{mod}\,q)+k-xrM^{q-2}(\mathrm{mod}\,q)})(\mathrm{mod}p)](\mathrm{mod}q) =$

$= [g^{k}(\mathrm{mod}p)](\mathrm{mod}q) = r.$

4.14. Схема идентификации Schnorr - Shamir.

Пусть n большое нечетное число. Выбирается случайное число k так, что НОД$(n,k) = 1$. Вычисляется

$h = -k^{-2}(\mathrm{mod}n) = -(k^{-1})^{2}(\mathrm{mod}n).$

В этой схеме n, h открытый ключ, k- секретный ключ. Алгоритм подписи сообщения $M<n$ выглядит следующим образом.

1. Выбирается случайное число r, НОД$(r,n) = 1$.
2. Вычисляется

$s_1 = \frac{1}{2}[\frac{M}{r} + r] \ (\mathrm{mod} n)$.

3. Вычисляется

$s_2 = \frac{k}{2}[\frac{M}{r} - r] \ (\mathrm{mod} n)$.

Подписанное сообщение (M, s_1 , s_2).

Проверка подписи осуществляется следующим образом. Вычисляется

$u = [\, s_1^2 + h s_2^2 \,] (\mathrm{mod} n)$.

Если выполняется равенство $u = M$, то подпись подтверждена.

Доказательство.

$$[\, s_1^2 + h s_2^2 \,] (\mathrm{mod} n) = [\, \frac{1}{4}[\frac{M^2}{r^2} + 2M + r^2] -$$

$$- \ (k^{-1})^2 \ \frac{k^2}{4}[\frac{M^2}{r^2} - 2M + r^2]\,] (\mathrm{mod} n) = M (\mathrm{mod} n).$$

4.15. Схема аутентификации Feige – Fiat – Shamir.

Это протокол аутентификации с нулевым разглашением. V, P - участники протокола, P доказывает V, что он P. P выбирает модуль $n = p \cdot q$, где p, q – большие простые числа. На практике $n \geq 512$ бит. Затем выбирается ν такое, что $x^2 = \nu \bmod n$ и существует $\nu^{-1} (\mathrm{mod}\, n)$. Здесь ν - открытый ключ. Тогда вычисляется наименьшее s, для которого $s^2 = \nu^{-1} (\mathrm{mod}\, n)$ (т.е. $s = \sqrt{1/\nu} \ (\mathrm{mod}\, n)$). Число s – секретный ключ. P надо доказать, что ему известно s, не раскрывая секрета.

1) P выбирает случайное число $r < n$ и вычисляет $y = r^2 (\bmod n)$

$$P \xrightarrow{\ y\ } V$$

2) V выбирает случайный бит b,

$$V \xrightarrow{\ b\ } P$$

3) Если $b = 0$

$$P \xrightarrow{\ r\ } V$$

если $b = 1$

$$P \xrightarrow{\ z = r\sqrt{\frac{1}{\nu}} \ (\mathrm{mod} n)\ } V$$

4) Если $b = 0$, V проверяет, что $y = r^2 (\mathrm{mod}\, n)$. Если $b = 1$, V проверяет, что $vz^2 = y(\mathrm{mod}\, n)$, что доказывает то, что P знает $\sqrt{\frac{1}{v}}$.

Если бы P знал бит b до своего первого шага, но не знал бы секрет $s = \sqrt{1/v}\ (\mathrm{mod}\, n)$, то то он смог бы обмануть V следующим образом.

При $b = 0$ P посылает V на первом шагу число $y = r^2 (\mathrm{mod}\, n)$, а на втором шагу он посылает число r. Тогда V убеждается, что $y = r^2 (\mathrm{mod}\, n)$. Если $b = 1$, то на первом шагу P посылает число $y = \frac{r^2}{v} (\mathrm{mod}\, n)$, а на втором шаге P посылает число $z = \frac{r}{v} (\mathrm{mod}\, n)$. В этом случае V проверяет

$$vz^2 (\mathrm{mod}\, n) = v\frac{r^2}{v^2} (\mathrm{mod}\, n) = \frac{r^2}{v} (\mathrm{mod}\, n) = y .$$

Таким образом, не знающий секрета P должен выбирать одно из двух различных сообщений до того, как он узнает какое из них потребуется. Если он делает это случайно, то вероятность успеха равна 0.5. Тогда за t циклов повторения протокола вероятность случайной аутентификации равна $(0.5)^t$, что при больших t делает процедуру аутентификации достоверной.

ЛИТЕРАТУРА

[Ама.] Амамия М., Танака Ю. Архитектура ЭВМ и искусственный интеллект. М.: Мир, 1993.

[Гилл] Гилл А. Линейные последовательные машины. Анализ, синтез и применение. М.: Наука, 1974.

[ГОСТ 94] Процедура выработки и проверки электронно-цифровой подписи на основе асимметрического, криптографического алгоритма. ГОСТ 3410-94. М., 1994.

[Гэ.] Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М., 1982.

[Кол.] Колчин В.Ф., Севастьянов Б.А., Чистяков В.П. Случайные размещения. М.: Наука, 1976.

[Ре.] Рейнгольд Э., Нивергельт Ю., Део Н. Комбинаторные алгоритмы. Теория и практика. М.: Мир, 1980.

[Хоф.] Хоффман Л. Дж. Современные методы защиты информации, М.: Сов. Радио, 1980.

[Ш.] Шеннон К. Работы по теории информации и кибернетике, М.: Иностранная литература, 1963.

[Biham 93] Biham E., Shamir A. Differential Cryptanalysis of the Data Encryption Standard, Springer- Verlag, 1993.

[Biham 94] Biham E. Cryptanalysis of Multiple Modes of Operation//Proc. ASIACRYPT'94, Springer- Verlag, 1994, c. 278-291.

[Biham 98] Biham E., Knudsen L. Cryptanalysis of the ANSI X.9.52 CBCM Mode/Technion – Computer Science Department – Technical Report CS 0928, 1998.

[Gol.95] Golic J. Intrinsic Statistical Weakness of Keystream generators//Advances in Cryptology – ASIACRYPT'94, Lecture Notes in Computer Science, Springer- Verlag, 1995, v. 917, c. 91-103.

[Goll.] Gollmann D., Chambers W. Clock Controlled Shift Registers: a Review//IEEE J. Scl. Ar. Commun., 1989, v. 7, N 4, c. 525-533.

[DSS] Digital signature standart (DSS). FIPS PUB 190, 1994, MAY 19.

[Massey 94] Massey J. Cryptography: Fundamentals and Applications. Advanced Technology Seminars, 1994.

[Matsui 93] Matsui M. Linear Cryptanalysis Method for DES Cipher//Pre-Proceedings of Eurocrypt'93, 1993, c. 112-133.

[Meier 89] Meier W., Staffelbach O. Fast Correlation Attacks on Certain Stream Ciphers// J. Cryptology, 1989, v. 1, N 3, c. 159-176.

[Men.] Menzes A., van Oorschot P., Vanstone S. Hadbook of Applied Cryptography, CRC Press, 1997.

[Mey.] Meyer C., Matyas S. Cryptography: a New Dimension in Computer Data Security. John Wiley & Sons, 1982.

[Ru.] Rueppel R. Good Stream Ciphers are Hard to Design. ICCST, Zurich, 1989, c. 163-173.

[Schn.96] Schneier B. Applied Cryptography Second Edition: protocols, algorithms and source code in C. John Wiley & Sons Inc., 1996.

[Schnorr 88] Schnorr C. On the Construction of Random Number Generators and Random Function Generators//Advances in Cryptology – Eurocrypt'88, 1988, c. 225-232.

[Simm. 94] Simmons G. Proof of Soundness (Integrity) of Cryptographic Protocols// J. Cryptology, 1994, v. 7, c. 69-77.

СОДЕРЖАНИЕ